Title A structured approach to malware detection and analysis in digital forensics investigation

Name Saeed AlMarri

A STRUCTURED APPROACH TO MALWARE DETECTION AND ANALYSIS IN DIGITAL FORENSICS INVESTIGATION

by

Saeed AlMarri

A thesis submitted to the University of Bedfordshire in partial fulfilment of the requirement for the degree of PhD

University of Bedfordshire

Institute for Research in Applicable Computing

April 2017

Author's Declaration

"I, Saeed AlMarri, declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research."

Title: A structured approach to malware detection and analysis in digital forensics investigation

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have cited the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help; and
6. Where the thesis is based on work done by myself or jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

A STRUCTURED APPROACH TO MALWARE DETECTION AND ANALYSIS IN DIGITAL FORENSICS INVESTIGATION

Saeed AlMarri

ABSTRACT

Within the World Wide Web (WWW), malware is considered one of the most serious threats to system security with complex system issues caused by malware and spam. Networks and systems can be accessed and compromised by various types of malware, such as viruses, worms, Trojans, botnet and rootkits, which compromise systems through coordinated attacks. Malware often uses anti-forensic techniques to avoid detection and investigation. Moreover, the results of investigating such attacks are often ineffective and can create barriers for obtaining clear evidence due to the lack of sufficient tools and the immaturity of forensics methodology. This research addressed various complexities faced by investigators in the detection and analysis of malware. In this thesis, the author identified the need for a new approach towards malware detection that focuses on a robust framework, and proposed a solution based on an extensive literature review and market research analysis. The literature review focussed on the different trials and techniques in malware detection to identify the parameters for developing a solution design, while market research was carried out to understand the precise nature of the current problem. The author termed the new approaches and development of the new framework the triple-tier centralised online real-time environment (tri-CORE) malware analysis (TCMA). The tiers come from three distinctive phases of detection and analysis where the entire research pattern is divided into three different domains. The tiers are the malware acquisition function, detection and analysis, and the database operational function. This framework design will contribute to the field of computer forensics by making the investigative process more effective and efficient.

By integrating a hybrid method for malware detection, associated limitations with both static and dynamic methods are eliminated. This aids forensics experts with carrying out quick, investigatory processes to detect the behaviour of the malware and its related elements. The proposed framework will help to ensure system confidentiality, integrity, availability and accountability. The current research also focussed on a prototype (artefact) that was developed in favour of a different approach in digital forensics and malware detection methods. As such, a new Toolkit was designed and implemented, which is based on a simple architectural structure and built from open source software that can help investigators develop the skills to critically respond to current cyber incidents and analyses.

Acknowledgement

First, I would like to express my sincere gratitude to my advisor, Dr Paul Sant, for his continuous support of my PhD study and related research, and for his patience, motivation and immense knowledge. Your guidance helped me throughout the research and writing of this thesis. I cannot imagine having a better advisor and mentor for my PhD study.

To my second supervisor, Dr Gregory Epiphaniou: Although face-to-face meetings could be counted on one hand, your guidance was invaluable. Thank you for your time and effort.

To the Director of the Digital Forensics Department of the Dubai Police, Captain Rashid Ahmad Lootah: Your encouragement and support have been unforgettable. You have played a significant role in my professional success and always encouraged me to learn and improve, especially in the field of security and digital forensics.

Last but not least, I would like to thank my family: Thank you to my parents and to my brothers and sister for supporting me spiritually throughout the writing of this thesis and in my life in general.

Table of Contents

List of Tables

List of Figures

Glossary

Notation	Description
API	Application Program Interface
APT	Advance Persistent Threat
AV-TEST	Name of a leading international IT security and anti-virus research company
C&C Arch	Centralised Command and Control Architecture
CIA	Security triad – Confidentiality, Integrity and Availability
CMS	Cash Management System
CORE	Centralised Online Real-time Environment
Digital Forensics	A way to analyse digital media using reliable information that can be applied to the investigation of an incident
DNS	Domain Name Server
FS-ISAC	Financial Services in Information Sharing and Analysis Centre
GUI	Graphical User Interface
HIDS	Host-Based Intrusion Detection System
IBM	International Business Machine, a leading IT organisation
ICAT	International Centre for Automotive Technology

ICS	Industrial Control System
IDS	Intrusion Detection System
IRC	Internet Relay Chat
Malicious	Unwanted program that can create a threat to the user or system
MD	Malware Detector
NDIC	National Drugs Intelligence Centre
NIDS	Network-Based Intrusion Detection System
NSRL	National Software Reference Library
P2P	Peer-to-Peer network model
PRM	Primary Research Method
RAM	Random Access Memory
RTS	Real-Time Scenario
SQL	Structured Query Language
TCMA	Tri-CORE Malware Analysis
TCP	Transmission Control Protocol
UBS	A multinational diversified financial service company based in Switzerland.
USSS	US Secret Service

CHAPTER 1 - INTRODUCTION

1.1 Overview

Malware is an intrusive malicious code that accesses computer system information without permission from the client. The role of malware is to infect computer network securities by obtaining unauthorised access to the network and distributing the malicious codes in the network. Currently, the internet is facing several problems related to malware, which ultimately hampers the security of computer networks. Malware is considered a global threat that has worked its way into all known parts of the worldwide computer industry. According to the Internet Security Threat report (by Symantec) published in 2016, over nine mega breaches were reported (a breach with over 10 million records is considered a mega breach), where total exposed identities by malware jumped 23% to 429 million that year.

The malware industry is growing and compromising computer securities across the globe by penetrating internet securities. Malware is not only damaging computer network systems; it is also a business entity for making profits. The developers of malware intrusion systems can potentially earn considerable amounts of money by developing malicious entity codes.

The role of malware developers is to develop malicious codes and transform them into malware programs. Malware developers do not create new suspicious malware each time; rather, they create new variants of existing malware and spread them throughout the internet shadow economy (Schipka, 2007). Variants of existing malware can be easily spread across internet securities. As stated in a report by Kaspersky Labs (Kaspersky 2014), there were 2,205,486 variants collected during an analysis of malicious activities across the globe in 2015 Q2.

1.2 Definition of Malware

Malware is software that includes instructions for performing unwanted actions in computer networks by placing vulnerable malicious codes across them. Malware is a combination of the words ‘malicious’ and ‘software’, which describes intrusive activities in the computer environment in computer terminology. Malware codes usually obtain access without user consent and damage the computer network by spreading malicious codes (Kramer & Bradfield, 2008).

1.3 Problem Overview and Motivation

Malware is intrusive software that conducts unauthorised, malicious activities on a system. It can be easily deployed into any system, causing longer disruption and damage. Malware poses a real, extensive threat, and it has created a buzz in the field of digital forensic science (Kramer & Bradfield, 2008). In the most pertinent situations, malware targets and attacks financial institutions or the banking industry, as well as other systems dealing with monetary assets both online and offline. Cyber criminals can create a script that targets customers and enterprises dealing with these financial institutions, creating a sense of insecurity amongst users (Yin et al., 2007). A comprehensive list of different types of malware would include Trojans, viruses, worms, rootkits, botnets, phishing, spam, spyware, keyloggers and logic bombs.

Traditional countermeasures used for detecting malware include signature-based methods or recognising anomalous behaviour specified by a rule or a function (Florencio & Herley, 2011). However, these traditional methods have flaws and challenges, thus making it difficult to obtain effective and efficient results during an investigation. Given the significant increase in malware attacks and exposure resulting in loss of confidentiality, integrity and availability of data (Florencio & Herley, 2011), the need for a new formalised approach to collect, analyse and mitigate malicious content in systems must be addressed.

Several reports have mentioned various problems addressed by malware investigators while detecting malware itself or related malicious activities. Some of these are as follows:

a) There is always a challenge in gathering and analysing large chunks of data for obfuscated malware or malicious code, as these procedures are usually time-consuming and costly and require different techniques (BITS, 2011);
b) Detection becomes complicated when an illegitimate activity employs sophisticated tools and methods (Schweitzer, 2002; Ye et al., 2010);
c) Added complexity arises when the same data are present over a large network or in various computational systems, especially when the detection is performed with pre-existing forensic tools, pre-defined malware detection systems and limited durations at workstations (Han et al., 2009); and
d) When numerous systems or parties are involved in the same crime, the analysis of independent digital data during the investigation can result in the loss of indispensable correlated evidence. This occurs due to the incapacity of digital forensics to craft correlations between multiple malware cases. It is exceptionally difficult to detect malware and to obtain accurate data because of numerous obfuscation techniques used by the programmers (Hunt & Slay, 2010).

Such problems can be addressed by focussing on various digital forensic techniques currently used to extract, gather and analyse malware-infected systems. Moreover, there is the opportunity for a new framework technique that focuses on typical features of malware activity, and which is capable of accumulating sufficient forensic evidence against the perpetrator and formulating it for presentation in a court of law. This framework should be able to optimise the needs and requirements of a robust investigation, and it should eliminate the associated challenges in malware detection and digital investigation to provide a cost-effective approach in the field of digital forensics.

1.4 Research Aim

This research was focussed on designing a framework that can help digital forensics experts to detect and analyse malware, and address the current challenges faced by investigators. The framework design is intended to help forensics experts carry out a quick, investigatory process to detect the behaviour of malware and its related elements. The proposed framework will also aid in assuring system confidentiality, integrity, availability and accountability. To propose the new framework, comparative research and analysis was undertaken using existing tools to address existing deficiencies in malware detection; however, to ensure that the analysis is comprehensive, the framework works with both manual and automated processes to detect threats and warn the system.

1.5 Research Objectives

The goal of the research was to optimise the research paradigm for malware analysis and to improve the investigatory experience by employing an active approach to detect malware programs. With this in mind, the researcher divided the objectives into two phases: Phase 1 focussed on developing a robust framework for malware analysis and detection, while Phase 2 involved developing an artefact as a prototype for an anti-malware Toolkit to assist investigators in systematic detection and analysis.

The objectives for Phase 1, developing a new framework, were as follows:

a) To conduct an extensive literature review of the different trials and techniques associated with malware detection and to identify the parameters for developing the most efficient tools for a live system;
b) To conduct extensive market research to understand the precise nature of the software/system requirements. Conceptual and contextual methods were used to gather information, followed by an extensive analysis of the report; and

c) To generate a new, effective framework to analyse the behavioural characteristics of the malware family and the mechanics of its operation. The complete observation of malware was exhibited in a controlled environment.

The objectives for Phase 2, developing the Toolkit, were as follows:

a) To develop a malware detection Toolkit, comprised of a simple structural prototype built from open source software, to help investigators develop skills to critically respond to current cyber incidents and investigations. The development should be reflected by the information gathered from Phase 1; and
b) To perform extensive testing of the Toolkit by experienced end users to gather real-time execution results.

1.6 Research Questions

The following research questions were set:

a) Is it possible to develop an effective malware detection system by investigating the behavioural characteristics of existing malware?
b) How can we develop a quick, inexpensive investigatory process to cultivate the parameters for developing the most efficient malware detection tool?
c) Is it possible to identify traces of malicious code using a three-phase (three-tier) methodology against the sophisticated data stored on multifunctional machines?
d) Is it possible to deploy a solution in the form of a framework suitable for all dynamic environments in the field of malware detection and analysis?
e) Is it possible to justify both static and dynamic methods in anti-malware technology?

1.7 Structure of the Thesis

In this chapter, the context, problem statement and aims and objectives of this research have been clearly presented.

Chapter 2 covers the literature review, whereby it discusses and justifies the scholastic work undertaken by various authors in the field of malware, including methods of detection and analysis. This chapter contains the current deficiencies, bottlenecks and challenges in running the current frameworks and tools. Chapter 2 also illustrates the needs and challenges faced by investigators in the field of malware detection.

Chapter 3 defines the research methodology for conducting the study and analysis. It identifies the precise problem statement for the study undertaken, presents the associated gaps in the research, and elaborates on the research objective by using a distinctive research methodology. Various methods and processes were selected based on the current requirements, objectives and problem statement discussed.

Chapter 4 defines the market research and analysis. Two different approaches were selected in the chapter – a questionnaire and interviews. The questionnaire was designed with a view to developing a new framework for malware detection and analysis, which involves a novel approach to digital forensics and research. The interviews were conducted with a view to gathering results for developing an artefact, or Toolkit, necessary for implementation and testing.

Chapter 5 defines the new framework for malware detection as well as a comprehensive explanation. This chapter describes a new approach (model) called the triple-tier centralised online real-time environment (tri-CORE) malware analysis (TCMA) method, which was developed as part of the current research and, as such, is a contribution to the field

Chapter 6 defines the artefact development (Toolkit). Within this chapter, it is the author's intention to develop an artefact that reflects the critical functionalities of the proposed framework, as outlined in Chapter 5 – a malware analysis Toolkit, which is simple in structure yet sufficiently robust using integrated open source malware detection software tools.

Chapter 7 discusses the results and observations on the testing of the Toolkit. This chapter illustrates the testing attributes used to deploy a successful functional Toolkit. Moreover, it presents the theoretical and practical implications faced during the implementation and testing phases.

Chapter 8 evaluates and concludes the research. It provides a summary of the findings and quantifications for completing the defined aim and objectives of the research.

CHAPTER 2 - LITERATURE REVIEW

2.1 Overview

This chapter clarifies and presents the most recent scholastic work done by various researchers in the field of IT security and malware's detection, analysis and associated software. The articles presented in this chapter were selected from reputable journals and conferences. The intention of the author was to first collect information on malware investigative procedures, methodologies on detection, current malware tools and related implications; then, study it thoroughly; and finally, analyse it. Next, the results of this process were to be utilised to develop a robust and reliable framework, providing a novel approach to the field of computer forensics and security. Gaining insights into the malware industry provides a better understanding of the current environmental scenario in areas of concern and associated potential risks. Special consideration has been given to the malware detection architecture in order to gain a clear picture of related pros and cons of both the static and dynamic methods. During the study, some practical implications were also highlighted, such as the difficulty with implementing some current methods/tools in malware detection techniques.

2.2 Introduction

With the increase of internet and technology usage, users are finding that they need to secure their valuable data and information in a protected way. The increased use of sharing confidential content through the internet potentially opens a door for hackers to steal protected data. Once a digital crime is committed, it is often then up to digital forensics experts to intervene. Part of a digital forensics team's responsibility is to monitor digital crimes occurring online. Forensics teams tend to follow a step-by-step methodology to detect online crimes, such as data theft and then provide evidence accordingly (Kent et al.,

2006). Experts can use contemporary information technology (IT) systems to investigate cybercrime. Conversely, attackers use malware, which can be spread widely to target computing devices holding valuable data. If the data is encrypted, attackers try to decrypt the data. Usually, malware embeds unethical code with the purpose of retrieving useful content from computing devices. However, it is often difficult to identify malware containing suspicious code designed to infect master machines (Casey, 2011). This is because such code is built using standard protocols and encrypted using complex techniques, making it hard to detect.

Although criminals can exploit the computational power of IT systems, forensics experts can do so as well, a discipline known as digital forensics. Forensics experts use this computational power to improve the efficiency and effectiveness of forensics, which helps in preventing and detecting malware activities and related infections from associated systems. For a profound analysis, it is necessary to understand the importance of digital forensics (John, 2012).

Digital forensics can be defined as a way to analyse digital media using reliable information that can be applied to the investigation of an incident. Computer forensics analysis, also termed *digital discovery*, refers to the process of examining digital-related evidence. An analysis is performed on the evidence that was taken at the time of the crime. A computer forensics department performs the collection, analysis and investigation of computer-related evidence (Tsai et al., 2012).

To make digital forensics more effective and efficient, it is crucial to understand the crime scenario, including both the event and the agents involved in the situation. Many crimes involve controlled mechanisms created using malware tools working against a common adversary, thereby generating intentional or unintentional extended attacks and threats. During the investigation, it is vital for investigators to understand the behaviour of the malware being used for these purposes (TzeTzuen et al., 2012).

2.3 The Global Impact of Malware on the Digital World

The expansion of the global digital world has seen a tremendous increase in the number of threats and exploited vulnerabilities. The number of malware attacks is increasing every day (Casey, 2011; Kent et al., 2006), which makes detection and analysis difficult. As shown in Figure 1 (AV-Test, 2015), in a comparison between new malware and total malware in a particular year, the global growth of total malware has increased over the past decade. This figure highlights the clear trend where a year-on-year doubling, or greater, in the growth of malware can be seen. This means that each year, new malware was added, and shows that malware reached approximately 400 million in 2015.

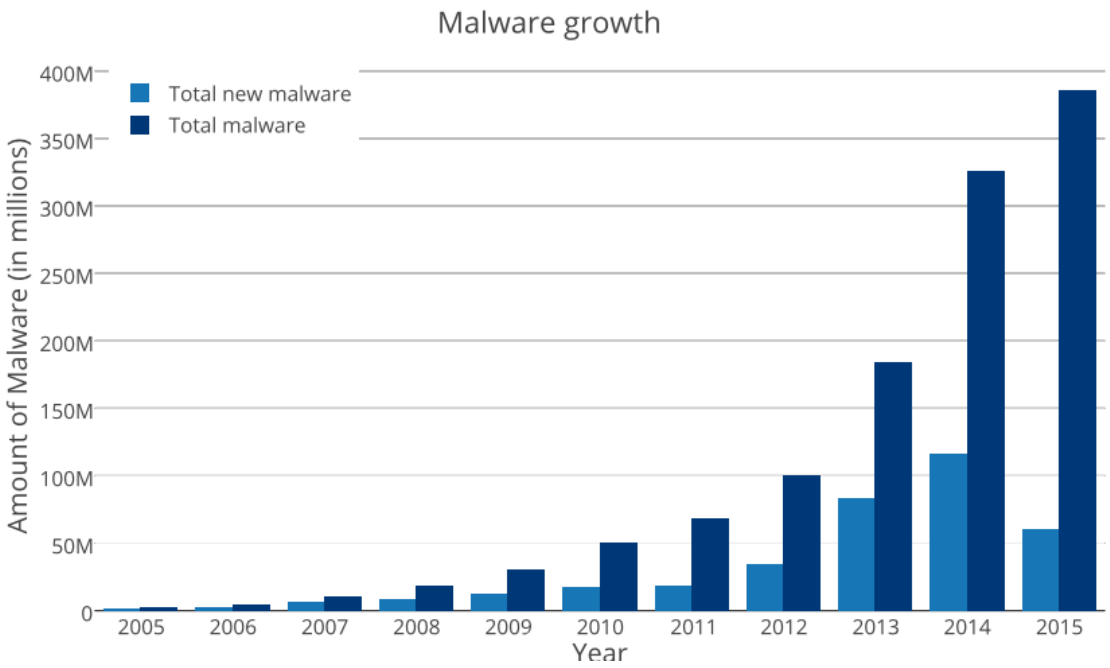


Figure 1 The Growth of Malware from 2005 to 2015 Globally (AV-Test, 2015)

When examining the motivation in relation to this growth in malware attacks around the world, attacks occur for a variety of reasons, but those related to financial gain are most prevalent (Sheriff, 2017). The increase in malware attacks correlates with the on-going global growth in internet usage (Figure 2), which, in turn, has grown from the increased use

of personal computing and the reliance the global business economy now has on the use of computers. Add to this more recent developments gaining popularity, such as using smartphones to pay for online merchandise or services and cloud services for personalised storage. As these practices have led to increased access to financial information used in everyday life, so has the opportunity for criminal activity to take advantage of this (Kaspersky Lab, 2014).

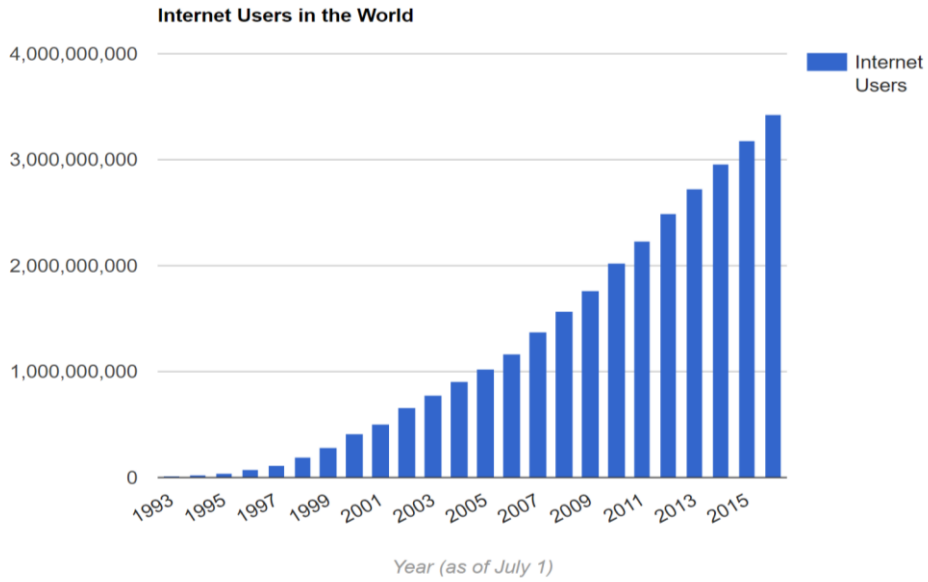


Figure 2 The Growth of Internet Usage from 1993 to 2016 Globally (internetlivestats.com)

As the business model of using malware is not a legal one, it makes tracing monetary values difficult to assess in terms of financial impact, despite data showing this trend of increasing growth. Reliable figures from malware sources are not readily forthcoming with their generated incomes, and companies affected by malware are not readily forthcoming with, or even able to, provide information on their losses (Intel Security, 2014). However, estimated figures of the cost to worldwide business in excess of $1 billion have been attributed to cybercrime as a whole with malware being a significant contributor (McAfee, 2013).

An examination of some of the more high-profile malware attacks that have occurred indicates that malware is more widely spread than just for financial gain. Aside from

sabotage or purely malicious intent, political reasoning (GhostNet and Red October malware attacks) is a common theme, as is environmental reasoning (Shamoon and Slammer malware attacks), which were the alleged motivations for targeted distributions of malware (Poulsen, 2013; Wangen, 2015).

With the reliance on computer systems throughout modern civilisation, the vulnerability to threat exists. The aforementioned attacks, and ones like them, have the ability to identify and target such computer systems with malware and the damage done can be considerable.

Harmful repercussions of a malware attack are not only limited to downtime losses or the loss of sensitive information within the system being attacked. An erosion of society's trust in the internet economy can be considerable, although somewhat immeasurable (Whittaker 2016), suggesting that a broad social impact can be just as damaging from any attack.

2.4 Malware Threat and Exposure Analysis

Malicious software impacts various sectors of society and is typically utilised by vindictive parties for personal reasons. Malware empowers individuals who seek to jeopardise digital forensics in financial services and engage in criminal activities. Malware inventors may often benefit financially from their programs, but on occasion, they may just want to disrupt business operations. One case study of malware used for disruption was the use of a Logic Bomb attack, reportedly used by a disgruntled computer administrator, resulting in damages in excess of $3 million to his company, UBS Systems. He was ultimately convicted on evidence from security analysts for implementing unethical security strategies and deliberately laid down the financial budget of the company upon the activation of the malware (SANS 2007).

There is an on-going and rapid increase in *Advanced Persistent Threat* (APT), which is defined as a class of malicious activities affecting a growing number of government organisations and business associations (Symantec, 2016). As described in the report of the Financial Services in Information Sharing and Analysis Centre (FS-ISAC), APT refers to a

secretive intent to engage in constant, relentless oversight of an individual, organisation or remote country, state, government or military in an attempt to perform malicious activities (DTCC, 2014). The report shows that an increasing number of APT attacks have taken place since 1986. The major threats involving APT agents are attempts to access and extract data that comprise delicate and/or gathered information (DTCC, 2014). Data comprising a variety of information types may be extracted, including innovative operations, licensed innovation, exclusive business forms, business techniques and/or information relating to individual officials. APT allows criminals to gain illicit remote access to host systems, including system maps and software alterations. Accessing the systems helps criminals to understand the business strategies of an organisation, thereby laying the foundation for possible illegal activities. APT consists of progressively malicious system tools that have been specifically developed to cause problems for organisations and individuals. For instance, Stuxnet and other malware were used to assault Iran's atomic plants, which may have given political as well as financial advantages to the intruders (Websense, 2014).

Various reports relating to on-going attacks on governments and business organisations across the globe have been published (Christie, 2006; DTCC, 2014; Websense, 2014). APT activities are carried out to aggravate IT systems, illegally access remote systems, and gather sensitive information. The use of spear phishing, an email spoofing fraud, is considered an essential methodological approach targeting governments and business ventures. The payloads of spear phishing have a direct impact on large business ventures, ultimately resulting in attacks on the business' operations. Often, malware utilises the most likely code to take advantage of an organisation's vulnerabilities; however, APT activities capitalise on new custom codes, which go unnoticed to large businesses and interfere with or antagonise systems. Engineers associated with APT are aware of loopholes in an enterprise's information security and utilise illicit codes to disable security measures of intrusion detection systems (IDSs), antivirus software and other security standards, thereby hampering the security of business enterprises (DTCC, 2014; Websense, 2014)

Some malware is particularly designed and implemented to disturb the operations of industrial control systems (ICSs). The Siemens ICS provided solid proof of the worst effects of the Stuxnet worm in 2010, which resulted in crushing sequences affecting the physical assets of major industrial solutions, such as gas pipelines, water conveyance and server farm environments; furthermore, it resulted in the distortion of industry frameworks. The major motivation of this malware was to interrupt the Siemens ICS's involvement in atomic projects in Iran. The features of the worm generated serious concern for operations and budgetary administrations (Falliere et al., 2011; Matrosov et al., 2011). With Stuxnet, the source code could be adjusted to focus on a more extensive scope of control frameworks in any number of discriminating industrial sectors. As previously mentioned, the budgetary administration segment has been targeted by ICS malware agencies with the particular intention to exploit its vulnerabilities (Falliere et al., 2011)

Data theft may also be performed to reveal information concerning innovative technology and strategic business operations. The main aim behind implementing such unlawful activities is to understand the business strategies and sell the company's research to its competitors, thus providing a financial incentive/payback. It is apparent from the examination of past and current cybercrimes that the financial industry has been targeted by malware (Baker & Goudie, 2010). A high profile case of this concerned SolarWorld AG, a manufacturer of solar products and the largest solar manufacturer in the United States (US), who was an alleged victim of Chinese state-sponsored cyber theft when financial, production, research and trade-litigation documents were stolen and handed over to Chinese state-owned businesses (Wilson, 2017). The case resulted in the US Department of Justice filing 31 criminal counts against five Chinese military personnel (Mickolus, 2015).

The US Secret Service (USSS) is an essential organisation in the examination of assets for the US Department of Treasury as well as its secret resources. The USSS research authorities have concluded that the financial sector has been severely affected by malware. It collaborated with Verizon and shared cybercrime case reports with them to facilitate the

cooperation of the two organisations. The idea behind this synergy was to efficiently manage cybercrime threats (Baker & Goudie, 2011). The confidential case report detailed affirmed security loopholes in firms that were either Verizon customers or part of the investigative ward of the USSS. Budgetary administration firms were the essential focus in 2009 and in 2010; thereafter, from 2013 to 2016, the emphasis shifted to the entertainment and public industries. Breaches related to the entertainment and public industries automatically affected the financial sector, as they disrupted the interactions associated with customers and reputation, resulting in financial losses. Regulatory controls were also suggested that involved new structures for transaction reports and risk management in financial sectors (Baker & Goudie, 2010, 2011).

2.5 The Malware Industry

Threats related to malware are typically controlled by criminal groups with organised plans of action to automate cybercrime. Software companies have developed strategic plans to exchange, introduce and remain informed about malicious codes to obtain benefits; however, the malware business has generated transmission and support systems to help criminals utilise malware successfully (Casey, 2011; John, 2012). Developers of crimeware profit from the sale or lease of malware to third-party vendors who then use it to carry out identity theft operations and account fraud. Figure 3 (Matrosov et al., 2011) illustrates the connection between segments in a common crimeware plan of action. Groups of criminals facilitate these tasks and the product is crimeware as a service (CAAS) (Perdisci et al., 2008).

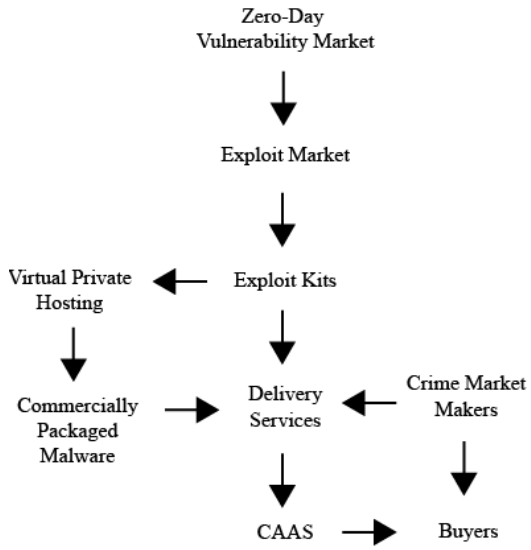


Figure 3 The Malware Industry Process (Matrosov et al., 2011)

The malware industry process itself illustrates the existing sources of threat that need to be scrutinised by digital forensics investigators for effective research. The procedure portrayed in Figure 3 begins when programming vulnerabilities are sought by offenders in a systematic manner. It starts with ‘zero-day’ vulnerabilities, which allow a vulnerable market to be identified, indicating greater potential profits for malware inventors (Coviello, 2011). These vulnerabilities are sold to criminals in the second step. Criminals engineer malware in an endeavour to render systems defenceless and create a range of different malware programs, which capitalise on different weaknesses in packs with parts that can be methodically introduced.

An example of this process was seen in the case of iFrame (Jakobsson, 2008), a web service technology that assists in creating a specific brand of malware. It allows a user to place a URL on ‘Server A’ that shows the content of ‘Server B’. Although this is normally done by legitimate users who want to show web content from multiple web servers on their server, criminals can take undue advantage of this feature by exploiting the web-based vulnerability in the iFrame, and placing their server content and links on the legitimate

server. As of 2016, iFrame was still vulnerable and an active exploit (iFrame HTML Security, 2016).

Several vulnerabilities are still possible in an unorganised framework long after they have been identified and exposed by cyber experts and investigative consortiums. Exploratory units may incorporate blends of zero-day and more seasoned assaults. The packs are arranged to send gathered information to private facilitators and administrations, and this design may be tweaked for a given purchaser (IBM, 2013). Malware creators contact potential clients by means of communication media, such as email. They connect with malware inventors to generate malware for clients who pay the business sector producers through covert ecommerce instalment frameworks. Crimeware operation is illicit, yet the danger of criminal indictment for individuals is minimised by the general industry process. In placing malware, each attacker's benefit involves introducing malware components to obtain monetary remuneration by exposing a general commercial domain to it (IBM, 2013). Several academics have investigated vulnerabilities to malware in different organisations (Provos et al., 2007). Moreover, scholars have composed papers on building and identifying endeavours that expose a company's vulnerabilities, which are not viewed as criminal activity (ethical hacking) (Jamil & Khan, 2011)

The relative danger of indictment for this sector is evaluated in Figure 4. It depicts the relationship between profitability and prosecution risks for those participating in crimeware activities. As each task is relatively isolated and only part of the complete product, the exposure each entity has to potential legal action is, therefore, relatively small. It is not until the latter stages of the malware process are reached that the risk of prosecution is high, but as these stages are reached, the potential to make substantial profit is also encountered, thereby making the business model an attractive proposition.

Figure 4 Relative Risk Compared with Profit Resulting from Participation in Crimeware Activity (Jakobsson & Ramzan, 2008)

2.6 Categorising Malware

Many different types of malware have been specifically developed, although some have even been developed unintentionally (Rieck & Duussel, 2008). Some types of malware present in the cyberworld will be briefly discussed with consideration of their impact and the destruction they cause to a system. Such malware requires special consideration and research, as these are the most common prevailing sources of threat to any system. Understanding their existence can provide an edge in designing an effective malware detection and analysis framework.

Every threat has unique attributes that distinguish it from others, making investigations more empirical. Presently, malware is becoming more complex and highly technical, incorporating various new behaviours and characteristics (Rekhis & Boudringa, 2011).

a) **Viruses, Worms and Trojans:** In the field of software education, a *virus* is defined as a self-replicating piece of source code attached to other programs that usually requires user intervention to instigate it (Adleman, 1990). *Worms* are self-replicating

source codes that can not only infect a stand-alone computer but can also spread through entire networks (Chen & Ji 2007) without user intervention. Worms work by detecting and attacking existing vulnerabilities in the system, making them difficult to prevent and detect (Chen et al 2003). A *Trojan*, also called a *Trojan horse*, is a program created to be useful and harmless with an unwanted malicious program hiding inside. This is one of the most common types of malware found by investigators and is considered responsible for many forms of cybercrimes (Krishan et al., 2012).

b) Backdoor: A *backdoor* code is a type of malicious program that allows attackers to bypass all security measures and controls in the system, making it vulnerable and allowing the attacker to gain access without any authentication or valid authorisation. The backdoor program is installed by attackers for future unauthorised accessibility and malicious activities, making the system completely unsecure for users (Dilkina et al., 2009).

c) Spyware and Adware: Spyware and adware are malicious, unwanted programs that perform stealthy and undesired installations and modifications to a system with malicious intent (Schultz, 2003). For example, this type of program can show an advertisement or hijack sessions on an internet browser, limiting the sites by showing unwanted websites on the pages (Lavesson et al., 2011; Qing & Dinev, 2005).

d) Botnets: Amongst the different categories of malware threats, botnets are one of the most widely used for computer crimes. Botnets are groups of malware infected computers that can work together to infect other computer systems. Different types of attacks use botnet malware. Examples include the malware attacks on Estonian banks in 2007 and the attack on Georgian government sites by Russia during the Russian invasion in 2008 (Kozlowski, 2014; Stephen & Lee, 2008). A botnet is distinct from other types of malware because it is not an explicit malware program (Cooke et al., 2005).

2.7 Digital Forensics

Digital forensics and computer forensics both use specific procedures to carry out their investigations. Investigating malware through digital forensics requires following a set of basic principles and fundamentals. Digital forensics comprises investigations related to all digital media, including digital cameras, mobile phones and digital players. In contrast, computer forensics involves the analysis of digital devices linked to computers and is a subset of digital forensics as a whole. Although this thesis focuses on the threat of malware to computer technology and, therefore, concentrates on computer forensics, the term digital forensics is frequently used throughout in reference to the branch of forensic science that is examined. In addition, the literature regarding the digital forensics process has been discussed many times in workshops conducted by forensics research groups (Agarwal et al., 2011). The fundamental principles remain the same with respect to how criminal investigations are executed. In 2001, the digital forensics process was presented in digital forensics research workshops, in which experts focused on discussing the methodology of implementing the steps involved from identifying an incident to the process of drafting the final reports regarding the incident (Rieck et al., 2008). Digital forensics involves several internal technical activities at each stage in the investigation. Implementing these basic fundamentals is mandatory in investigative agencies and institutions that need to be compliant and adhere to the abovementioned specifications. The stages involved in the forensic process are shown in Figure 5 (Alink et al., 2006).

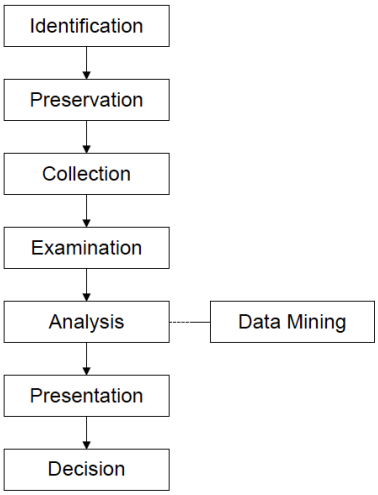


Figure 5 Stages Involved in the Forensics Process (Alink et al., 2006).

A brief discussion of each stage of this forensic process is provided as follows:

a) Identification

Incidents are detected in the identification stage. The incidents are acknowledged based on complaints through system monitoring or via any indications implying the need for investigation. This stage includes creating strategies related to how the investigation should be handled in the subsequent stages (Alink et al., 2006).

b) Preservation

Preservation is the first stage of digital investigations in which investigations are performed on digital objects before they have been handled for analysis. Case management tasks are involved in the preservation category. During this stage, imaging of the original data is executed via examining technologies (Alink et al., 2006).

c) Collection

Creating a replica of the original digital content is the primary aim in this stage. Authenticated hardware and software components are used to perform this task.

Collection procedures are executed while keeping in mind the investigations performed in the previous stages. Preservation steps are followed so that the integrity of data is maintained (Alink et al., 2006).

d) Examination

Examination tasks are applied to data collected in the previous stages. Numerous steps are carried out in the examination phase beginning with filtering the data. Filtering comprises specialised techniques based on database management. Several files in a computer belong to different specifications, including some for the operating system and some related to the software and the machine's applications. Digital forensics tools are synchronised with databases to extract the most relevant data from the machines. Investigated machines may be adjusted at different time intervals so that further procedures can be applied. Data from the machines may have been deleted to remove the evidence from the sources; and if this is the case, recovery procedures are executed. Data recovery performed on deleted data helps investigators to extract valuable content from the computer (Richard & Roussev, 2006). Methods used in the examination phase may result in changing the state of the data, resulting in a loss of the integrity of evidence to be presented in court; thus, step-by-step documentation is mandatory at every phase of the investigation to maintain the integrity of data. Encryption and decryption of the data are performed in the examination stage. Moreover, examinations are carried out on a copy of the original data, leaving the suspicious content untouched (Alink et al., 2006; Richard & Roussev, 2006).

e) Analysis

In the analysis stage, statistical procedures are employed and data extracted from these procedures are used in investigations. Analysis is required to gain in-depth knowledge about data objects and protocols, which are also linked. Techniques such as time-lining and synchronisation of data objects are commonly used for the analysis. To make

suspected data objects logical and comprehensible, data mining techniques are used to detect complicated patterns and relationships amongst different data objects. During this stage, data mining concepts are related to the digital forensics of the data and analysis reports are drafted for further investigations (Alink et al., 2006)

f) Presentation

In the presentation stage, experts write testimonials based on investigations performed in the previously described stages. Documentation is recorded and presented to legal authorities (Alink et al., 2006).

g) Decision-making

Decision-making is the final stage of malware investigation where presented reports and documentation are analysed by legal authorities. The decision-maker can be a judge or a jury, if the case is presented in a court of law. Corporate cases are related to corporate crimes and evidence is presented to corporate law authorities in the case of an internal breakdown of policies (Alink et al., 2006)

The flow of the process is important for the success of any investigation as it deals with a wider point of view concerning the crime scenario. It also assists with gathering evidence related to a complete event, examining the actual process of collecting evidence in a methodical and systematic manner. Digital forensics investigation is an integrated process that requires the collaboration of all stages to ensure effective and efficient outcomes.

2.8 Efficiency and Effectiveness in Malware Analysis

Efficiency and effectiveness are the major attributes of malware analysis and need to be maintained in each phase of an investigation. Effectiveness helps in identifying important aspects of the evidence, thereby resulting in meaningful investigations. Efficiency in digital forensics is related to resources that are used to gather the evidence. These attributes are

different in terms of their functionality and play independent roles in digital forensic investigations.

With malware developers employing different techniques to deter and hide from analytical tools in a bid to remain unseen (McAfee, 2017), along with the increase in malware being released (Symantec, 2016), the need to employ the appropriate methodology is paramount. Malware investigations can be performed using manual techniques as opposed to automated techniques, although these are often regarded as time-consuming in comparison (Ayers, 2009). Usually, the performance of an investigation deteriorates if manual techniques are carried out on large volumes of data due to the inherent inefficiency of this method (Manar, 2016). There is also the use of distributed and service-oriented architectures where evidence can only be tracked by correlating the utilised common communication. Appropriate tools are required to execute each of the digital investigation phases to obtain usable forensic information for further investigations (Vuong et al., 2011). Ayers (2009), who carried out research on Encase and the Forensic Toolkit, found that manual forensic tools (termed first-generation tools in his work) require a large amount of human monitoring, which impacts investigation speeds, costs and error-making. The major issue with implementing first-generation tools is that execution and processing speeds are very slow. Today's investigative scenario uses multiple detection methods, which take more time and require high-processing systems. Therefore, new methodologies and approaches are required for a state-of-the-art infrastructure as manual tools have (or will very soon) become obsolete. Thus, special consideration is given by the author in providing an approach that carries multiple detection methods on a single platform, keeping the pace and flexibility on par with conventional requirements.

Although large amounts of new malware are developed and released each year (Symantec, 2016), most of it is a variation on only a very few programs. Techniques such as polymorphic reproduction or random crypt seeds are used to produce slight changes to the binary code (Iliopoulos et al., 2011), but the underlying malware code remains largely

untouched. Analytical tools used to individually investigate each altered malware, albeit with slightly mutated binary code, results in inefficiency in its identification and capture. Developing solutions that do not individually analyse the same file, even though subtle differences may be present, and which determine a behavioural profile instead, result in a far more efficient methodology. This has been evidenced with early development already delivering considerable time savings (Bayer et al., 2010).

To improve the efficiency and effectiveness, data representation algorithms can be applied to present data so that they can be easily understood by investigators. For querying and indexing digital evidence, XML-based prototypes can be used. It is possible to embed data into XML using special tools, resulting in significant improvements in the querying process. Under this method, a querying language called XQuery is used to improve query handling (W3C-online). This provides special resources for investigators so that they can search the required data. The main goal of implementing such protocols in the research is to make investigators' work easy when dealing with large data volumes; additionally, the system also significantly improves the ability to detect relevant information from the evidence.

Maintaining data integrity and efficiency in digital investigations is crucial and needs to be maintained during each phase. Richard and Roussev (2006) focused on modern methods for improving digital forensics and reported that to achieve effective malware detection, data must be presented in a specific format and high-level data abstractions must be followed to generate significant evidence files. Avoiding customary files is suitable when analysing evidence targeting several files together. Modern improvements include writing hash signatures and using large-scale hash databases to encrypt data files with a better configuration system to perform encryption with less vulnerability.

To handle large volumes of malware, automated techniques have been developed, which rely on artificial intelligence to produce techniques that reason and learn (Han et al., 2009). However, as malware detection has become more advanced, the same is true for malware itself where techniques to evade automated systems have also been developed. This

development of evasive techniques has produced malware that requires run-time studying (dynamic analysis), so malware strain behaviour can be efficiently identified and mitigated. This behaviour analysis is a more detailed analysis that is not necessarily detected and resolved by a single tool (Rieck, 2011).

Therefore, the automation for the detection and prevention of malware requires a new framework; the author of this research proposes a novel framework that utilises an automated parallel methodology. Here, parallel methodology signifies a framework with multiple methods and tools in any investigative arena. Such a Toolkit is discussed later as an artefact for this research.

2.9 The Malware Detection Architecture

The architecture associated with malware detection illustrates approaches used for detecting bugs (malicious programs) in the system. Previous works describe two approaches for malware detection and analysis categorised into static analysis and dynamic analysis techniques. In publications such as Gross (2009) and Mukamurenzi (2008), the authors mention a hybrid approach, combining both static and dynamic techniques for malware investigation. Individual methods possess various advantages; however, combining both methods can lead to more effective and efficient malware detection and analysis.

a) Static Analysis Method

Static analysis examines the static disassembling code of a program without running any executable applications. Basic static analysis will consist of checking the functionality of a program such as its signature or behaviour. It is a simple and quick approach, although it provides limited information for malware analysis (Justin & Klein, 2011)

b) Dynamic Analysis Method

Dynamic analysis carries out an investigation by running the whole program, mostly in a virtual environment – for instance, performing experiments for checking API calls, instruction traces, looking for registry modifications, variability in system memory and so on (Alazab et al., 2010). In Rieck et al. (2011), the authors used the method of system calls to capture traces of malware based on its behaviour. Bayer et al. (2010) also propose a dynamic analysis technique, whereby the program traces executable logs, which are then analysed to check transition probabilities.

c) Hybrid Analysis Method

The hybrid method is an amalgamation of the static and dynamic methods. This method is discussed in Mukamurenzi (2008) and Bayer et al. (2010). The authors designed a framework called Malware-DNA, which uses the technique of a debugging-based behaviour pattern to monitor and perform assessments on the findings to confirm the presence of malware. Using both techniques gives the analyst an advantage by providing benefits from the fidelity of the dynamic analysis and the efficiency of using a static analysis. Additionally, a survey report by Shijo and Salim (2015) shows the results of an experiment where the accuracy rate was 95.8% using the static method, 97.1% using the dynamic method and 98.7% using the hybrid method. This shows a drastic improvement in the accuracy and completeness rate, as compared to individual static and dynamic methods. Combining both these methods can provide a stable yet effective result in malware detection, thus making it a reasonable underlying function in proposing a new framework for malware detection and analysis. The only drawback of this method is its slowness during malware investigation and analysis; however, this should not be a limitation for those users for whom efficiency outweighs the long timeframes involved during investigation.

2.9.1 Examples of Malware Detection Architecture

There are two different types of detection architecture, the first of which is a centralised command and control (C&C) architecture, and the second, a decentralised C&C

architecture. Determining which one to use is based on the way communication is implemented (Maheshwari, 2010).

A centralised C&C architecture approach will see a central server as a connection point that controls and monitors all activity. This approach is divided into two sub-groups: internet relay chat (IRC) where online communication (chat) occurs in real time; and HTTP-based where an IP address is used as a central server to communicate and send instructions. A decentralised C&C architecture approach is based on a peer-to-peer network (P2P) model, a control system that sees infected computers simultaneously act as malware and C&C servers. Commands are received and transmitted to each malware item and do not rely on a single communication point to deliver instructions (Gupta et al., 2016).

2.10 Malware Detection Techniques

Various malware detection techniques can be used to discover vulnerabilities by examining patterns in malware. Different means and mediums are engaged in the detection of malware and each has a specific responsibility, depending on the type of evidence being sought. Digital forensics and malware detection teams have different roles and responsibilities, such as collecting evidence, performing analysis, creating reports as evidence, maintaining custody of documents, etc. In the following paragraphs, different types of common detection methods are presented and briefly outlined.

a) File System Analysis

File systems (or filesystems) provide a mechanism for users to store their data in files and directories. According to Carrier (2005), ‘File system analysis examines data in a volume (i.e., a partition or disk) and interprets them as a file system’.

A file system is where data is arranged within a computer system in a logical and ordered manner so that it can be found. Analysis of this file system is a commonly used process

within digital forensics as it finds information about system accessibility (Farmer & Venema, 2004). The process itself extracts layers of data from the computer for examination where the layers can be combined to discover any malicious files that are alien to that computer system. To escape detection, malware developers use special techniques to avoid affecting files, such as introducing encrypted code and links to malware files. File system analysis excels by being able to analyse data that is hidden behind data files, which would otherwise go unnoticed, and access deleted files which may be used to trace files stored in different system locations (Rahmatian et al., 2012). *Due to its importance, this method is key in the proposed solution*.

b) Analysing Keywords and Identifiers

A simple technique to extract information regarding malware is to use a keyword search tool. *For the proposed solution,* it can help in identifying files associated with the malware. Keyword searching is an easy way to evaluate bot master communication threats. Ample critical information, including domain names, internet protocol (IP) addresses and strings, can be used for analysis by investigators using this method (Edem et al., 2014). Another technique that can be integrated is to use a 'string search'. This search is helpful in determining the malware's attributes related to string values. An example of a string search is the string 'key-log', which can be used to fetch data related to strings. Sometimes, this may return incorrect results because other tools, such as antivirus software, may contain the same string values in their databases, thus creating contradictions. Searching through strings can be accomplished in many phases of the investigation and is generally carried out when expert knowledge is required for the investigation (Edem et al., 2014).

Investigators can conduct an analysis on a forensic image using keyword searching. To obtain optimum results through searching, it is very important to understand the level of searching. Searching is performed in a systematic way and passes through a level of abstraction, thereby revealing information on which the search is performed. Identifiers and

strings are considered to be the best sources for detecting malware on host machines. A variety of sources can be searched to detect malware threats, such as intrusion detection system (IDS) logs, firewall logs and the information extracted from previous forensic analyses.

c) Detection Using Intrusion Detection Systems

To detect anomalous activities, a special detection system is used that identifies inconsistent activities and is known as an IDS (Siddiqui et al., 2008). Improvements can be made in detecting intrusions by implementing data mining techniques that can automatically interrogate large data sets, thereby reducing time and increasing efficiency (Nadiammai & Hemalatha, 2014). IDSs are used to detect malicious codes on suspected machines. There are two types of IDS domain:

- The host-based IDS (HIDS); and
- The network-based IDS (NIDS).

Signature techniques are used to identify different malware and are the essence of IDSs. These are anomaly detection systems and are responsible for detecting anomalous activities. Moreover, investigators use advanced techniques by employing data mining methods. HIDS systems are used globally as a malware detection technique and are integrated with intrusion detection tools. Host-based detection systems are rich in their functionality as they can analyse program code signatures and clarify the behaviours of malicious activities. The responsibility to detect malware intrusions lies within both network- and host-based detection systems.

For the proposed solution, the system can be preloaded with IDS capabilities for detecting botnet intrusions. IDSs can be used to detect malware intrusions, which can be analysed using digital signatures in the proposed solution. These techniques will be used to obtain

information regarding the botnet architecture and a host of information regarding botnets, which can be used for detailed botnet analysis (Burji et al., 2010).

d) Data Mining Techniques for Malware Detection

Data mining has been a primary focus for many researchers in the field of malware detection and analysis. It is considered an analytical tool for analysing infected data and summarising the findings in a meaningful way. Reports generated from data mining are useful in the process of analysing and detecting malware, and can later be presented as evidence. Currently, data mining is used in digital forensics to provide solutions and implement concepts to evaluate the relationships between internal factors related to unauthorised access to files, executable links, anomalous behaviours within the system, and external factors related to exploitation by compromising the confidentiality, integrity and availability of stored data (Amos et al., 2013). The motive of integrating data mining functionality in the proposal is to gain insight into a large amount of data to find traces of the injected malware (Amos et al., 2013). Data mining concepts will be used to uncover unsuspected relationships between the data, and to summarise the content into useful and easily understood information. Investigators may gather variable data patterns belonging to different files and systems, which can be combined to generate useful information that can later be formed into a statistical report to reach a final conclusion (Thuraisingham, 2011).

The techniques of data mining will be useful in understanding large datasets by analysing attributes and extracting important facts related to a specific malware program. Mining techniques play an integral role in the detection of malware during forensic analysis (Bhavani, 2009). To improve the structure of datasets, data mining is integrated with data warehousing and data with specifications on malware is stored in the same database. Integrating data mining and data warehousing could be helpful in extracting important information on injected malware and the results can be retained for later use in investigations. The efficiency of the data mining operation results relies on how the data is

evaluated and represented in investigation studies (Duhan et al., 2009). Data mining techniques can be categorised according to different specifications, depending on the objectives with which the extractions are performed. Various modelling techniques (mentioned in Szor, 2005) can be used in data mining, according to the sources used. Two common techniques will be used in the proposed solution:

i. **Descriptive Modelling:** Descriptive modelling can be used to increase knowledge regarding existing patterns of malware present in the system. This modelling technique involves discriminating and summarising existing explored data from the infected system. Common examples of descriptive modelling include segmentation and cluster analysis techniques, which are considered the best techniques for malware detection and analysis under descriptive modelling (Szor, 2005).
ii. **Predictive Modelling:** Predictive modelling is performed on the outcomes of known results and is used in making predictions regarding existing malware and other malicious programs in the system. During investigations on malware, predictive modelling attributes are combined with descriptive modelling. Descriptive techniques are used before making any future predictions regarding the information. Examples of predictive modelling include rule-based analysis and regression techniques, for which the detection rate is extremely high (Szor, 2005).

For the proposed solution, the researcher can use the dataset as the input for initialising the technique. At the beginning of the process, data are collected from the infected sources and represented as objects. Objects are analysed by software that measures their attributes and features. To achieve efficient results through the data mining process, it is necessary to select representative data for the input process and then perform pre-processing accordingly. To obtain the right form of data, transformations are carried out on the datasets,

which convert data repositories into important information. In the case of the artefact Toolkit, we have integrated basic fundamentals of data mining while searching the artefact, as we are collecting data repositories of malware types and categories during investigation, which can be searched for better utilisation and converted into datasets through the data mining process.

e) Attack Vectors Detection

An attack vector can be defined as a technique that involves gaining access to infected machines. For this research, it is vital to start the process by accessing an alleged machine and performing analysis and associations on typical vectors that are linked with peripherals connected to the machine, such as CDs, DVDs and other removable media. Due to advances in technology and widespread internet connectivity across the globe, attack vectors have become established in the communications domain. The most common activities associated with internet attack vectors are search engine manipulation, Trojans, social engineering and P2P file-sharing activities (Carrier, 2005). By using the associated tool in the proposed artefact, attack vectors can be detectable and analysis can be done at any level of investigation.

f) Identifying the Characteristics of Malware

To carry out the malware detection process, it is essential to collect information on as many attributes related to the evidence as possible so that relevant details are easily found. Evidence is analysed and important malware infection patterns are used to detect malware intrusion. This method will help uncover malware evidence through an iterative process; it includes systematic, step-by-step detection techniques that can be integrated into the ***proposed framework***. In each phase, evidence is gathered and analysed by investigators.

g) Timestamp

To detect malicious and irregular activities caused by malware, timestamps were applied in the proposed framework. Integrating anti-forensics techniques with timestamps can be a slower process, as anti-forensics attempt to counter-attack investigation techniques, which can produce a negative effect on the evidence and the quality of an investigation. During the process of transmitting C&C information between an infected host and a bot master, timestamp information regarding irregular activities is generated. Anomalies can be detected and monitored using network monitoring tools. Timestamp information can also be employed during the post-mortem analysis of stolen log files (Wolff, 2015).

h) Detection through the Hashing Technique

It is also possible to use hash signatures, a method based on signature detection techniques and considered one of the best forms of encryption (Clarke et al., 2012), which store the malware file codes in hash databases. The National Drugs Intelligence Center (NDIC) hash keeper, hash sets and the National Software Reference Library (NSRL) reference datasets are examples of hash signatures. Search techniques play an important role in the filtration of the data, as desired files can be sought from full-length codes. These searches can be easily and directly performed on live servers or on suspected hard drives (Shannon, 1948).

i) Analysing the System Configuration and Settings

Investigators can analyse the applications installed on a computer to investigate suspected malicious behaviours. The Windows registry plays a vital role in revealing malware code files on a computer. Thus, it is important to include a registry investigation in the proposed solution. The registry keys hold an ample list of executable paths. To carry out investigations on installed applications, the following registry key can be checked: HKEY_LOCAL_MACHINE\Microsoft\Windows\CurrentVersion. Trace files for installed applications can also be analysed by investigators to identify activities related to malware. It

is necessary to have some Windows registry datasets that store suspicious trace files and can be used to detect malware traces. Malware has the greatest impact on the registry of a computer and its functions by adding the malicious content to auto-run registry settings (Nguyen et al., 2010). Investigators can also obtain information regarding installed applications by examining pre-fetch files, which are automatically created when a program is executed in Windows. Pre-fetch files can help extract start-up information when a program is launched on the Windows platform. For an in-depth examination of user and system actions, event logs can be used; these keep track of user activities and reveal critical and non-critical activities. To access the event viewer, it is necessary to develop a functionality that triggers a command, that is, eventvwr.msc (Nguyen et al., 2010).

2.11 Challenges in Malware Detection

During the literature research, many difficulties and challenges of malware detection were found throughout the various sources. For a consistent yet effective framework, it is important to address all the challenges. These challenges are mentioned and discussed as follows:

a) Time-consuming and Complex

The actual spectrum of malware threats is wide and complex, especially if costs related to system damage are included. No computing base or network platform is immune from such damage. In traditional malware detection, methods were based on an antivirus system that considered malware to be a virus or a worm and used detection mechanisms based on signature-based algorithms or the acknowledgement of a heuristics approach based on specialised behaviour. Modern malware is multi-partite in nature, and thus incorporates several multiple infections and payloads in a single instance and situation (Duhan et al., 2009). Traditional means of detecting malware are failing due to the large customisation involved in detection frameworks, making them rapidly obsolete. Moreover, various anti-forensics techniques have been deployed to create a bottleneck in terms of detection, and

which delay the ensuing removal of infections from a system. The increasing use of the entropy method and extensive knowledge of malware makes detection more complex and time-consuming (Szor, 2005).

b) Changing Behaviour Heuristics

Improved malware types have a tendency to change their behaviour heuristics to go un-noticed as an increased level of avoidance. Using automated systems, internal code can change and appear different to any detection method, resulting in a missed flagging of the malware or what appears to be a continuous stream of viruses attacking the system, thereby eventually flooding it. A variety of different adaptive techniques can be used to accomplish this, and this method is being used more frequently (Kolosnjaji et al., 2016).

c) Poor Reference and Documentation

Studies on malware analysis have provided large amounts of data that are often too conflicting to provide any definitive result. Moreover, data is not always documented in detail for referencing purposes. A complete rationale is given in an article by Szor (2005), focusing on the criticality in malware detection and analysis. Szor (2005) argues that most malware analysis companies specialise in antivirus algorithms and tools; therefore, they do not want to share their experiences or knowledge in this field to protect their proprietary information (Neelakantanand & Rao, 2008).

d) Problems with Traditional Methods

Most companies follow ‘signature-based’ methods to recognise virus-infected programs. Signatures are simply detected patterns in the form of hash-containing bytes of malware in an executable format. They are collected in a known database and often used together with algorithms or methods of detecting suspected files (Demme et al., 2013). Problems associated with this method (mentioned in Vinod & Laxmi, 2009) are that malware tends to change its functionality or characteristics automatically, often making it difficult for any

antivirus software to fully detect it. For this reason, variations in scanning for malicious programs are required.

e) Variable Malware Techniques

Malware infections are gradually evolving through variable techniques. Generally, malware exploits vulnerabilities existing in any given software; however, with new software versions, there is the possibility to introduce new patches. Thus, dynamic development must take place in the field of malware analysis to meet the challenges in frequently used software (Vinod & Laxmi, 2009).

f) Challenges in Referencing and Linking

Investigators are often unable to create matching patterns linking previously recorded malware data with new malware data. This implies that existing infected files may have been corrupted by other malware. When this occurs, the investigator's result will be inaccurate, and malware detection and analysis are less likely to be provided (Demme et al., 2013). In some scenarios, costs and skills related to existing malware detection create a challenge for investigators, especially when investigators and analysts lack knowledge in the malware detection domain.

g) Difficult Data Mining Process

Many pitfalls can occur during the data mining process. Some of these possible pitfalls mentioned in Milenkovic and Jovanov (2005) are as follows:

- As systems will contain data with different formats and specifications containing both infected and non-infected files, inconsistencies may arise in the accuracy of the results, and vast amounts of data with variable formats may harm the execution if not properly implemented; and

- Meaningful results from data mining can only be justified if systematic processes are followed. In any situation in which an investigation is carried out using static analysis of malware files, valuable information may be obtained in a fragmented format. Thus, through data mining, various results could be obtained that do not provide insights into the mechanism of malware development (Milenkovic & Jovanov, 2005).

To address the abovementioned challenges and issues, the author will be introducing new methods of proliferation and malware data remediation (in Chapter 5) to check the probability of the occurrence of malware reinfection. New methods and techniques will be used to update the traditional framework and incorporate them into the new one. The new methods are intended to increase the efficiency and effectiveness of malware detection and analysis. For instance, using a hybrid analysis model (using both static and dynamic methods) reduces the implications and associated challenges, making the framework more robust and effective.

2.12 Summary

This chapter dealt with the aforementioned related literature by various scholars and publishers on the subjects of IT security and malware. The author has tried to select relevant material related to the problem statement and the objective of the research (mentioned in Chapter 1). The chapter is written in the context of the issue of malware and its detection to better understand potential solutions in this field. The chapter was initiated by discussing the increase in the dynamism of technology and its data and security using sophisticated tools and techniques. It was found that cyber criminals are taking advantage by using the internet to steal confidential data and sharing it with the public, thereby harming the integrity and availability of these sources. Thus, to highlight the situation, it was vital to discuss the responsibility of experts to investigate online crimes and provide evidence to protect internet users. This discipline is widely known as digital forensics. It was also explained how experts

use computational power to improve the efficiency and effectiveness of forensics to detect computer-related crimes with relevant evidence provided.

A section on the increased number of threats by malware was discussed via statistics provided by AV-Test (2015), demonstrating global malware growth. It further explained the involvement of associated crimes using malware to work against a common adversary, creating extended, intentional or unintentional attacks and threats. It was suggested that malware was found to empower individuals who pose a danger to all online services, but especially within the financial services sector. Another section (2.4) discussed the on-going and rapid increase in advanced persistent threat (APT), referring to the class of malware affecting the cyber world. APT allows criminals to gain illicit remote access to host systems, including system maps and software modifications. Often, malware utilities represent the most likely code to take advantage of an organisation's vulnerabilities; however, APT activities exploit new, custom code not easily detected by large businesses, and which interferes with and antagonises their systems. Various vectors create infections from malware, and these are distinguishable based on the category in which they are included.

Some types of malware present in the cyber world have been briefly discussed here, as well as their effects and the damage they can cause to a computer system. These types include viruses, worms, backdoor programs, botnets, malicious codes and Trojans. Malware poses a large threat of loss to the industrial sector, particularly the financial sector. To control the level of infection, malware forensics and detection activities play an important role. A digital forensics process involves several internal technical activities in each investigatory stage performed. The categories used in a digital forensics process are Identification, Preservation, Collection, Examination, Analysis, Presentation and Decision-Making.

Also highlighted as important was having a detection mechanism capable of achieving effective and efficient investigative results. To do so, the importance of an efficient and effective malware detection mechanism was found to be a major attribute of malware

analysis that must be maintained at each stage of the investigation. Effectiveness helps to identify the important aspects of the evidence, resulting in meaningful investigations. Efficiency is related to the resources used to gather evidence. These attributes differ in their functionalities and play independent roles in digital forensics investigations. Various malware detection methods have been discussed in relation to the attributes that can be used to detect patterns of malware vulnerabilities in a system. Two major detection methods – a static analysis method and a dynamic analysis method – were discussed. A relatively new integrated hybrid technique was also mentioned to highlight the importance of embracing both static and dynamic methods as a proposed solution. In addition, techniques were identified in the use of digital forensics, including time stamping, attack vectors, data and link mining, hashing, data representation, keyword and identifier analysis, intrusion detection mechanisms and file and format analysis. These methods have justified an intrinsic, yet enhanced solution for digital forensics investigations. Although they simplify the processes used by investigators to generate conclusive reports, they still pose challenges during investigations, such as limited availability of forensics tools, conduction and poor database patterns used in matching and detecting malware, which are not often updated, and a lack of custodial data in the labs. Thus, this literature review not only affirms existing approaches regarding malware detection, but highlights the need to develop new frameworks and methods for detection and analysis, as discussed in Chapters 3, 4 and 5 of this thesis.

The final section acknowledges some of the common challenges in malware investigation and analysis, which once resolved, are to be part of a proposed solution in using a new framework for the problem statement (mentioned in Chapter 5).

CHAPTER 3 - RESEARCH METHODOLOGY

3.1 Introduction

This chapter defines the methodology for conducting market research as well as the analysis presented in the next chapter (Chapter 4). In this chapter, the author's intention is to present the associated gaps in the research and to elaborate on the research objective by using a distinctive research methodology. Various methods and processes were selected based on the current requirements, objectives and problem statement discussed in Chapter 1 (particularly Sections 1.3 and 1.6). Qualitative and quantitative approaches were implemented, whereby primary data were collected through a questionnaire-based survey and face-to-face interviews. Targeted samples of digital forensics professionals were asked to give feedback on selective questions that focused on issues related to the problem statement of the research. These selected questions were based on the problem statement and the objectives of the research. As this research needs more precision in subjective areas, it was considered important to get feedback from seasoned specialists. Thus, the entire methodology focuses on two methods – a questionnaire and an interview. In the questionnaire method, selected questions were based on the topics of malware, IT security and detection tools. The process was initiated by distributing the questionnaire via email or physical post to only those candidates comprising IT professionals in the fields of security and forensics, or professionals in the field of malware and related tools, followed by responses received for analysis. As the selected respondents did not represent the wider community or society affected by malware, the results obtained are of interest but cannot be generalised to cover the whole population. However, these selected respondents play an important role in their respective companies/countries, such as the United Arab Emirates (UAE), India and the United Kingdom (UK). For the face-to-face interviews, the respondents were selected on the basis of their professional experience working with specialised malware detection and analysis tools or their respective knowledge in the field. The rationale for selecting the respondents for the interview was as follows:

The process of selecting candidates was initiated by scrutinising the respective profiles of all the respondents participating in the questionnaire process. The set criteria are illustrated below.

a) Candidates should be seasoned specialists in the field of digital forensics or malware investigation; or
b) Candidates should possess hands-on malware investigative tools experience or have had experience in the field of malware investigation through extensive malware detection toolkits; or
c) Candidates are part of an organisation dealing with anti-malware tools and toolkits; or
d) Candidates work with any government organisation in the field of cybercrime and computer forensics.

3.2 Definition of Methodology

A research methodology determines the process model or plan for conducting research and collecting data for sound qualitative or quantitative analysis. The model deployed was divided into two phases: first, developing a new enhanced framework for malware detection and analysis, which is intended to be a novel approach towards digital forensics; and second, gathering results in developing an artefact that reflects a toolkit required for implementation and testing.

Selecting and implementing a methodology is important as it assists in collecting repositories of data based on facts and literature. It simplifies the process and assists in defining new observations and vital conclusions in a research project.

Three major examination approaches have been selected, based on the following research paradigms:

- Quantitative research
- Qualitative research
- Mixed research

3.2.1 Quantitative Research

Quantitative research is an examination standard that depends on the accumulation of numerical information gathered using quantitative or statistical techniques. In general, this type of data is a large chunk of information for analysis and provides a more specialised approach in the research (Marshall, 1996). For this research, a more generalised problem statement of ‘detecting and preventing malware activities in the user’s system’ exists, which requires a more formalised and wide approach of collecting statistical data from respondents to make analysis more viable for the larger audience.

3.2.2 Qualitative Research

Qualitative research is defined as a technique that depends on the accumulation of subjective information extracted from the methods and approaches used. This is a more generalised approach and information could be gathered from small, related respondents (Marshall, 1996). For this research, one of the major objectives is to design and formulate a new framework of detecting and analysing malware activities in an enhanced manner. The data needed to be collected from subjective professionals and seasoned specialists. This provided precise opinions from respondents in a collective manner.

3.2.3 Mixed Research

Research paradigms can be based on mixed research, which includes the blending of quantitative and qualitative methods. Mixed research can be further blended into different categories of mixed research, where each category may depict different research statistics. Numerous structures can be developed through blending qualitative and quantitative research attributes (Marshall, 1996). Furthermore, for this research, the mixed research approach has been deployed across many engineering disciplines by integrating the results

and data gathered from both the above methods. Blending and integrating the analysis results provided a more reliable and robust solution, which was later used for proposing a new framework for the research. Moreover, this integrated report assisted in developing the artefact for the study.

3.3 Data Collection

Various methods can be used for collecting, designing and conducting the study in the form of data. This section of the research focuses on the processes and procedures for research manipulation and management. Two major concepts are reflected by this methodology; these are related to issues that were highlighted within the literature review (Chapter 2) of this thesis, such as challenges in current malware detection, optimisation in malware detection techniques, awareness of malware-related threats among users, effective tools for the investigators, etc. The two selected methods for this research are the primary research method, often referred to as the current analysis technique, and the secondary research method, also known as the conceptual analysis technique.

3.3.1 Primary Research Method

The primary research method includes the original source of information, which is mainly extracted from the source data and is not analysed before being included in the assessment sheet. Assessment criteria play a vital role in the collection of important information from people through fieldwork. The essential information is regularly gathered through face-to-face meetings or conversations with the relevant community. Other mediums can also be used to extract the information, such as telephone communications, radio correspondence and email exchanges (Curtis, 2008). For the research presented herein, primary research plays an important role. A common theme identified in the literature review (Chapter 2) was that malware has a distinct dynamic nature, with new developments and issues occurring periodically, making it essential to perform data gathering and analysis. In this research, the

primary sources of data collection consisted of interviews and questionnaires, which will satisfy the current needs and requirements of the study.

The questionnaires were distributed through telephone communications and email confirmations to all the respondents in the UAE, India and the UK. In order to design an appropriate questionnaire, the work focused on the literature related to the study of malware analysis and detection tools. The design for these methods included patterns of collecting information based on the research questions of the study (Turner, 2010). For the interview part, questions were designed according to the research question, reflecting the problem statement of the research (Driscoll, 2011). Here, the interview process focused on the experience, recommendations and references from respondents associated with the field of computer security or related technologies. It also included understanding the scope of malware detection and analysis. The design of the interview questions was related to the subject of the research, and prior examination of the questions was performed by personnel working in the respective organisations. The extracted information was then evaluated and observations were made to draw conclusions. Information gathered from the literature review uncovered some important problem areas that needed to be discussed. These areas were considered vital and needed to be acknowledged during the primary research in the form of a questionnaire or interview questions. Some of the key areas are as follows:

a) Questions based on knowledge and opinions about malware;
b) Questions based on various security processes, adopted technologies and gained skills in malware detection;
c) Questions based on malware detection tools and analysis; and
d) Questions based on problems faced by investigators while conducting malware analysis.

3.3.2 Secondary Research Method

Secondary data can be defined as information that has been gathered via the analyst's research and which does not involve assessments from the fieldwork. The data extracted from the secondary sources was utilised in no less than one layer of examination before being incorporated into the needs assessment. This is required to achieve more precision and accuracy in the report. The secondary data can be categorised into different classes that may contain data with different specifications. Secondary data can be specified as web materials, communication media reports, published journals, acknowledgements and information that has been cleansed, broken down and gathered for a reason other than the needs assessment (Church, 2001). For the research presented, a mixture of information collected from genuine internet sources, such as IEEE, books and published documents, was selected. Based on previous work, the author selected the methodology to justify the findings of the study; however, for this section, research through current analysis was selected. This methodology involves certain design attributes such as explanatories, standardisation, methods for reaching the target audience, permissions, ethical considerations and the environment. This helped in determining the way in which interviews were conducted. It also constituted information authenticity, allowing the author to select relevant respondents and factors that reflected the changes in the results. This activity was performed by implementing a qualitative method. A qualitative method is explanatory in nature and this approach assisted with obtaining the best understanding of the problem statement to ensure a precise solution. It also helped the author by providing exposure to current needs and requirements, which was necessary to perform informed observations and evaluations in order to obtain a genuine and valid result for the research (Crabtree & Miller, 1992).

As highlighted above, the primary method was used to focus on collecting information and actual facts related to current market research. The study presented here used two different methods to conduct the primary research paradigm, namely questionnaires and interviews.

Designing the questionnaire assisted in assessing current data for the requirement of developing a new framework for malware detection and analysis, while the interview planning assisted in assessing the needs and requirements for the development and implementation of a Toolkit (artefact).

3.3.3 Questionnaire Planning

The questionnaire was designed by selecting respondents related to the field of IT and those who specifically have knowledge of the subject associated with the research. In order to design the survey, observations were referenced from Chapter 2 (literature review) and other related studies. It is important to match the design and planning with the aims/objectives of this research. All the standard practices must be followed, including the ethical considerations from pre-analysis to post-analysis techniques. The ethical considerations that were followed during questionnaire planning are as follows:

a) Confidentiality and privacy;
b) Including only willing participants;
c) Prior notifications and information on the survey's intent; and
d) Following privacy laws and regulations.

In addition, the whole questionnaire is divided into sub-categories to get a clear understanding and transparency from the feedback and observations at a later stage (Chapter 4).

The rationale for selecting a questionnaire as part of the research methodology is described below.

- *Defining the problem statement*: A primary research methodology will help in identifying the real problems faced by investigators during malware detection and analysis. Hence,

using a questionnaire assisted the author in defining the problem statement of the study in a much more realistic manner.

- *Understanding needs and requirements:* Studies that use a primary research method assist a researcher in understanding the needs and requirements of the end user. Quantitative and qualitative analysis can be used to determine the precise and accurate needs of all end users facing the same dilemmas. Questionnaire analysis gives statistical data that is more realistic and provides accurate information on the current needs and requirements of users of malware detection and analysis. In addition, bottlenecks in any malware detection process were acknowledged.
- *Discovering evasion techniques:* Conducting primary research for the study can provide information on various evasion techniques. An enhanced obfuscation applied to the malware code makes detection and analysis flexible and simplistic for an investigator. Thus, the gathered information helped in determining various recommended evasion techniques.
- *Analysing an attacker's perception:* A questionnaire can also help in identifying the perception of the attacker; this can be done by selecting questions on behaviour analysis carried out by investigators based on their experiences. This valuable information helped in determining the vulnerabilities of existing systems and assisted the author with determining underlying updated security features to be included in the proposed solution.
- *Analysing malware detection tools:* One of the major objectives of this research is to develop an enhanced detection toolkit. Thus, conducting primary research for the study provided statistics for various adopted malware programs in the market, which helped in building a comparable chart to create a report (part of the thesis) selecting malware detection tools for the proposed solution.
- *Verifying integrity:* To check the integrity of the proposed framework and methodology, it is optimal to perform a third-end analysis. A third-end analysis (primary research) provides a genuine analysis to determine whether the previous work done by the author

was accurate. Thus, required changes can be made at a later stage to avoid inaccuracies in the final results. For instance, much of the literature mentioned in Chapter 2 was cross-verified with the analysis report of the questionnaire.

3.3.4 Interview Planning

Interview planning was designed by selecting organisations in an associated field of information technology. The structural documentation contained pre-defined questions that were designed for face-to-face interviews. Various IT experts in the area of IT and security with knowledge of malware were selected as respondents. In order to design the questions, analysis of the collected literature described in Chapter 2 was used. This analysis therefore focused on the requirements of current and future malware investigative techniques, how these requirements are presently being met, and any limitations or potential flaws within the current set of available tools. Subsequently, the interview questions were emailed to the selected professionals one day before the interviews were conducted to provide them with a better understanding of the topic. Giving only one day was required ethically and also helped to prevent the respondents from providing influenced answers. Moreover, this activity helped them prepare their responses to the questions before the recording, thereby providing reliable and integrated results within the given timeframe. For efficiency, the interview questions were reviewed prior to the interview process to ensure clarity. An audio recorder was selected as an optional device for recording the answers for better understanding as well as relevancy of the answers in order to maintain quality. At the end of the interview, the respondents were also given an opportunity to read their own transcripts before they were finalised to ensure that they were satisfied with their answers on the topic.

The following steps were carried out before conducting the interviews:

a) The organisations were selected based on their industry and the resources they use to conduct their business. Other factors, such as employee ratio, tangible assets and turnover, were considered to check the appropriateness and genuineness of the

organisations for inclusion. Another reason to consider such factors is that companies with high employee ratios and high turnover are more prone to malware attacks and have significantly more exposure on issues related to IT security due to the large capacity of data (Kroll, 2016). Thus, to satisfy the research objectives, the above criteria were selected. The organisation needed to have recognised experience with an average amount of data on their servers.

b) A letter of permission asking to conduct the interviews was written to the managing directors of the organisations. Selected employees were also contacted via email to invite them to an interview. In the case of any cancellation, an alternative day was requested from each respondent.

c) Ethical concerns were kept in mind while designing all questions. It was ensured that no personal questions outside the related subject were asked, and if the respondents felt uncomfortable at any time, they were welcome to discontinue the process.

d) The interview began with basic questions, and gradually, questions on more detailed topics were asked that were more aligned with the aims and objectives of the research work.

e) The collected answers were monitored, evaluated statistically and further analysed to obtain meaningful results.

3.4 Data Assessment Technique

Upon collecting a large amount of data, assessment plays a critical role in defining conclusions and suggestions for further work. These assessments can be divided into personal observations and factual descriptions collected from the analysis. The assessments can be further sub-divided into different categories, including emergency assessments. An emergency assessment is an examination of destructive areas and is used to construct a mutual understanding of the impacted sources.

The collected data can be gathered from assessments, including the following options:

- Collecting reports regarding the critical impact on the subject;
- Producing discoveries regarding activities that are not definitely known;
- Gathering information which is collected for a dispute analysis;
- Investigating the impact of a change in a situation that specifically or indirectly resulted from the impact on the associated subject involved in the study; and
- Using a pre-characterised arrangement of research methods to guarantee effectiveness in information gathering and investigations.

Before designing or implementing a prototype, it was important to gather information by conducting an effective market analysis to identify the exact needs and requirements of the user, particularly users who were exposed to current software and any limitations that were present. For a more extensive analysis of the data and the question put forward, a primary research paradigm was executed. The primary research involved accumulating data by conducting interviews in which questions reflecting the problem statement of the research were selected. In this scenario, market research focused on obtaining recommendations from the experts and practitioners related to computer security, cybercrime investigation or malware detection methodologies. This subsequently led to conducting the research and an analysis was undertaken to establish the outcomes.

3.5 Summary

In this chapter, the author's intention was to elaborate on the research objective and identify the precise problem statement for the study undertaken, and the associated gaps in the research by using a distinctive research methodology. As the author selected qualitative, quantitative and mixed methodologies, respondents were chosen at distinctive levels for both the questionnaire and the interview. Special attention was given to ethical

considerations, such as privacy, confidentiality, legal feasibilities, etc., before implementing these methods. Furthermore, special criteria were executed for selecting the respective respondents for both the interview and the questionnaire. Precise planning was done beforehand, including the rationale for designing questions while doing cross-referencing and integration with the gaps analysed in the literature review (presented in Chapter 2). Gathering data posed some challenges, such as sending out the questionnaire in a variety of ways, such as email, telephone, post, etc., to all respondents. Special consideration was placed on the data assessment technique to achieve a quality analysis report for conclusion (see Chapter 4).

CHAPTER 4 - MARKET RESEARCH AND ANALYSIS

4.1 Introduction

Based on the selected methodology discussed in Chapter 3, the author divided the market research into two phases: a questionnaire and an interview. Selecting and implementing a methodology is an important task as it assists in collecting repositories of data based on facts and literature related to the aims and objective of the research. It simplifies the process and assists in defining new observations and vital conclusions in a research project. As the research proposes a solution to the given problem statement for a more generalised audience by means of experienced and specific respondents who were seasoned professionals, two different approaches were selected. The questionnaire was designed with a view to developing a new framework for malware detection and analysis, which involves a novel approach to digital forensics and research. The interview was conducted with a view to gathering results for developing an artefact, or Toolkit, necessary for implementation and testing. Each market research method illustrates responses and statistical data with analysis and a detailed description. The author also concluded the analysis to provide results in a more efficient way, which is used later in the chapter to formulate the framework and to develop the artefact.

4.2 Market Research (Phase 1): Questionnaire

During the initial research, the questionnaire was distributed via email to the target respondents. The framework used for the questionnaire was based on a ‘scale item’ approach, which is a method that uses scale analysis (in Sijtsma & Verweij, 1992) as a measure to assess data gathered from respondents. A pilot study was conducted to authenticate the questionnaire and 189 feedback copies were collected. Further follow-up was conducted at a later stage to get more responses; however, due to professional commitments, the respondents were not able to share their feedback within the allocated timeframe for this follow-up stage. The respective questions, feedback and

acknowledgements are provided in Appendix A.

There were four sections in the questionnaire comprising specific topics (mentioned in Appendix A). The sections were as follows:

- Section A: Knowledge and opinions about malware;
- Section B: Various security processes, adopted technologies and gained skills in malware detection;
- Section C: Malware detection tools and analysis; and
- Section D: Problems faced by investigators while conducting a malware analysis.

4.2.1 Reliability and Validation

The cross-sectional study consisted of data gathered by distributing a questionnaire to all candidates who were professionals working in the domain related to malware and its attributes, and who were based in the UAE, India or the UK. A core analysis was performed based on the experts' experiences, investigation cases handled and their designation in their respective organisations. The selection was also made based on industry standards, followed by the number of years the experts had spent in their respective professions. It was found that out of 189 candidates, 43% of the respondents were professionals from IT security, 33% were forensics experts and 17% were network administrators, while 7% belonged to other IT fields. In terms of experience, the largest group (33%) had 5–8 years of experience in the field of computer/digital forensics.

Table 1: Respondent Demographics

	Experience (Years)				
	0–1	1–3	3–5	5–8	< 8
Per cent	10	17	23	33	17

4.2.2 Discussion of the Results

This section illustrates the data analysis, which was based on the designed questionnaire conducted to formulate a design solution for the problem. The details of the questionnaire analysis are given below.

SECTION A: Questions based on knowledge and opinions about malware

Q-1	**What is your role?**			
Options	Forensics expert	Networking admin	IT security	Other
189	69	30	80	10

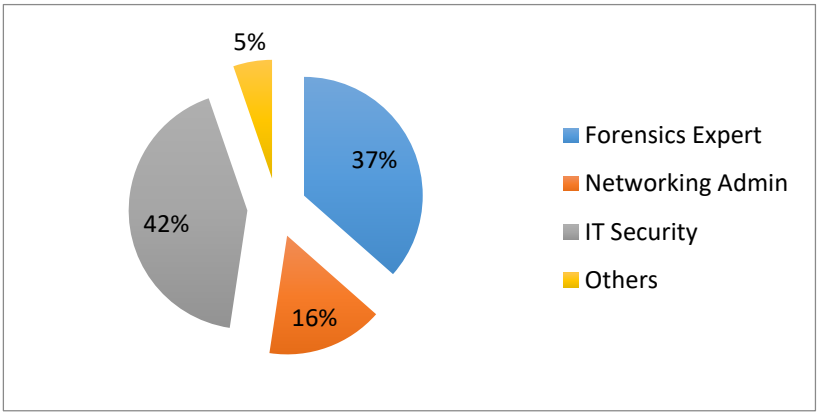


Figure 6 Tabulated & Graphical Results of Role of Respondents

Analysis: To conduct the analysis, feedback was gathered from 189 respondents. Of all the respondents, 42% were from IT security, while 5% belonged to other IT fields. This shows the integrity and authenticity of the study, as all selected personnel had direct relatable experience to the subject area.

Q-2	**How much experience do you have with modern malware?**				
Options	0–1 year	1–3 years	3–5 years	5–8 years	> 8
189	15	25	35	70	44

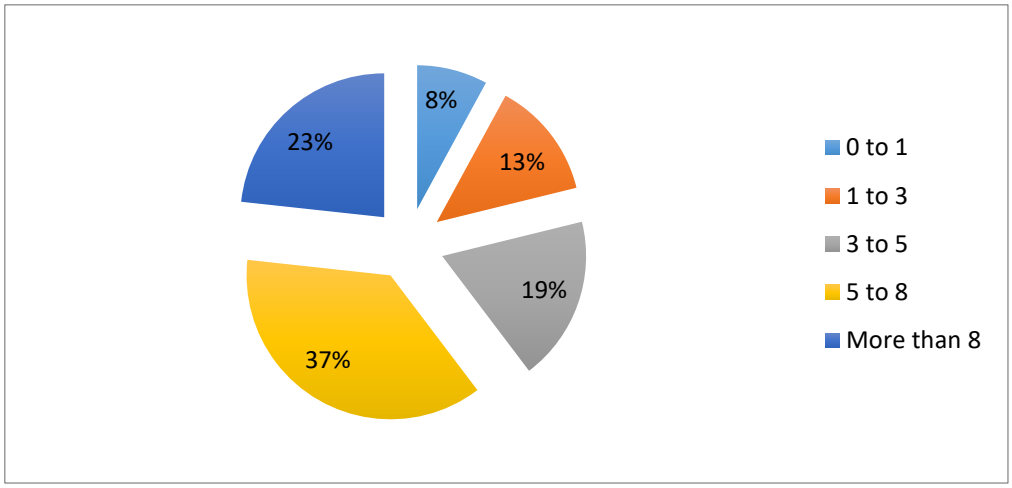


Figure 7 Tabulated & Graphical Results of Experience of Respondents with Malware

Analysis: The respondents making up the largest portion of the sample (37%) had 5–8 years of experience in their respective fields, while those with 0–1 year of experience represented the smallest portion (8%). These percentages emphasise the credibility of the feedback on the basis of experience. Formulating data from experienced professionals should provide support for the subjective accuracy of the survey.

Q-3	How often have you seen changes in the malware landscape?			
Options	Never	Seldom	Occasionally	Frequently
189	2	13	95	79

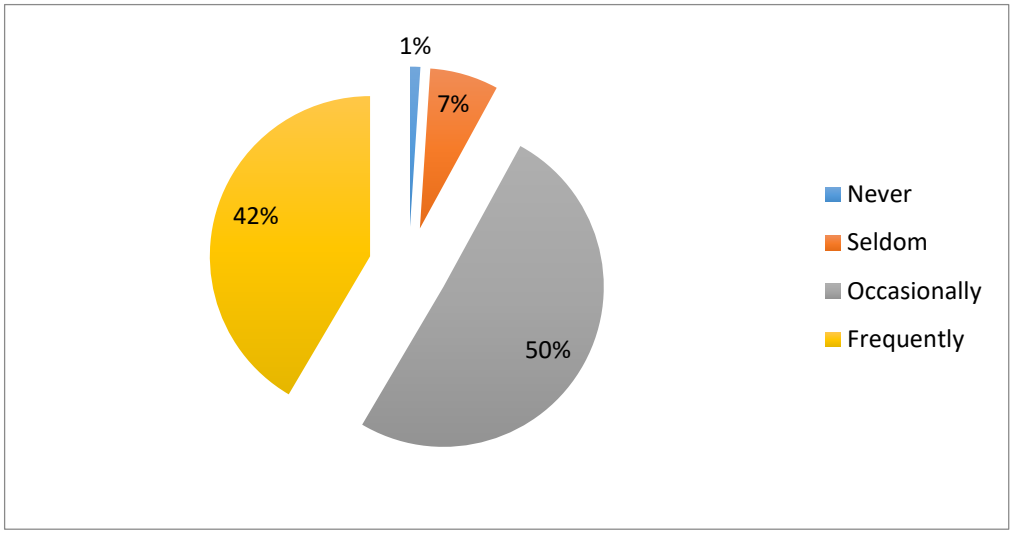


Figure 8 Tabulated & Graphical Results of Changes Seen in Malware Landscape

Analysis: The malware landscape is a major area of concern for any security professional. Changes in the malware landscape illustrate the modulations in the nature of malware attacks on systems. In this situation, a significant number (50%) of respondents perceived occasional changes. The view here of the respondents correlates with reported increases in malware attacks; for example, a 36% increase in malware variants was found from 2014 to 2015 (Richter, 2016). This is highly concerning, as malware detection becomes more difficult and complex for any tool with such a high ratio in modularity. This result helps to signify and validate the research question addressed in the following chapters.

Q-4	Is your organisation vulnerable to malware attacks?		
Options	No	Yes	Not sure
189	40	89	60

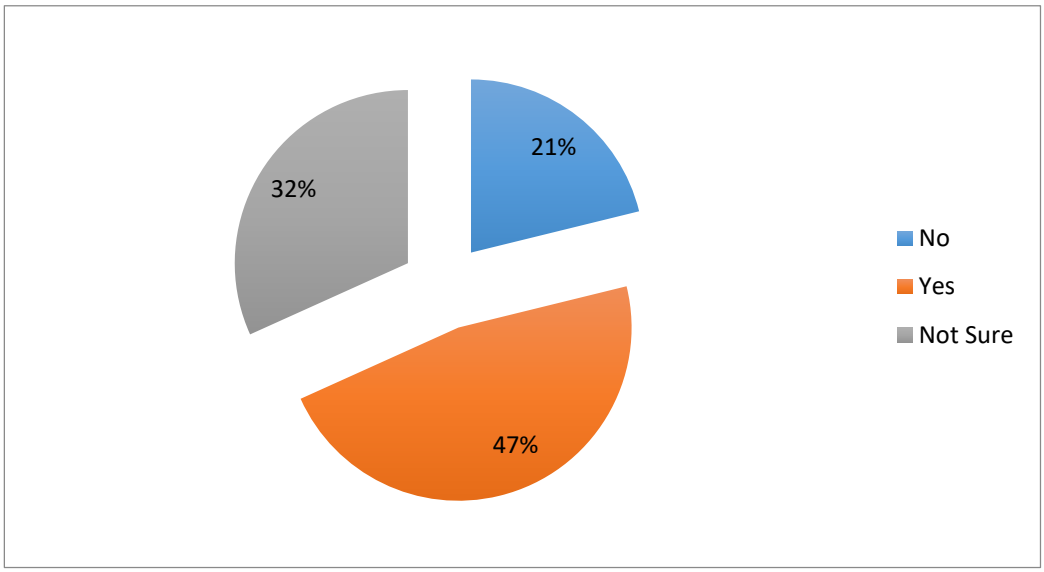


Figure 9 Tabulated & Graphical Results of Vulnerability of Organisation to Attacks

Analysis: Most respondents indicated a sense of vulnerability in their schema, which they considered prone to malware attacks; 47% accepted that their own organisation could be vulnerable to attack and then possibly open to infection by malicious codes. This result helps to show a sense of insecurity and a lack of confidence in the existing security protocols available in the experts' organisations.

Q-5	What is the situation for other non-technical employees? Are they capable of identifying a malware attack?		
Options	Yes	No	Not sure
189	70	110	9

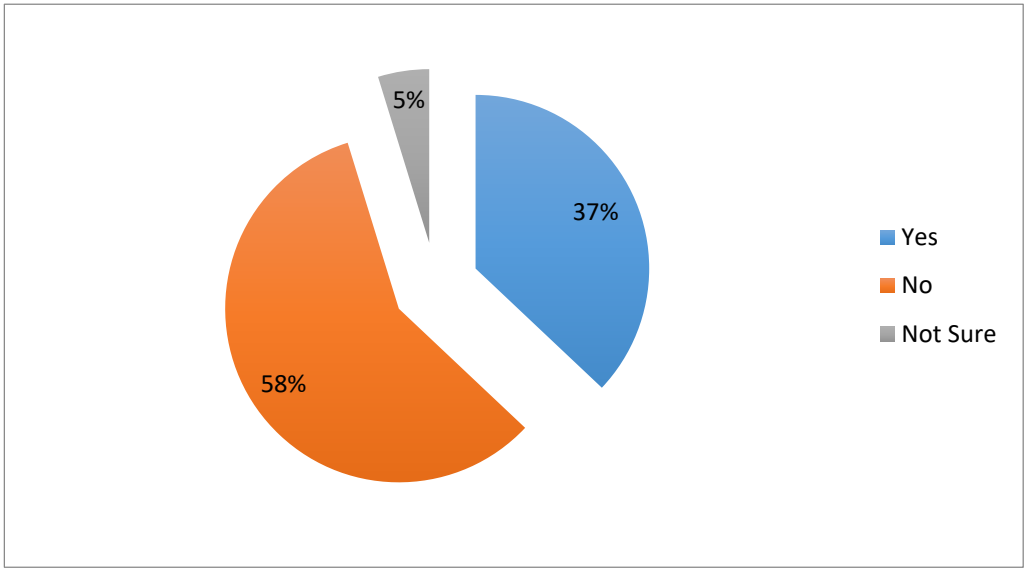


Figure 10 Tabulated & Graphical Results of Capability of Non-Technical Employees to Identify Attacks

Analysis: This result shows a perception of a lack of awareness regarding malware attacks amongst non-technical employees; 58% of the respondents indicated that most of the non-technical employees are not aware of malicious codes or malware. This suggests that the possibility of detection by employees outside of the specific area of IT professionals is insufficient; hence, there is a need for education or possibly an automated detection mechanism to provide assistance.

SECTION B: Questions based on various security processes, adopted technologies and gained skills in malware detection

Q-1	Do you have any expertise in the malware handling process?		
Options	Yes	No	Not sure
189	120	49	20

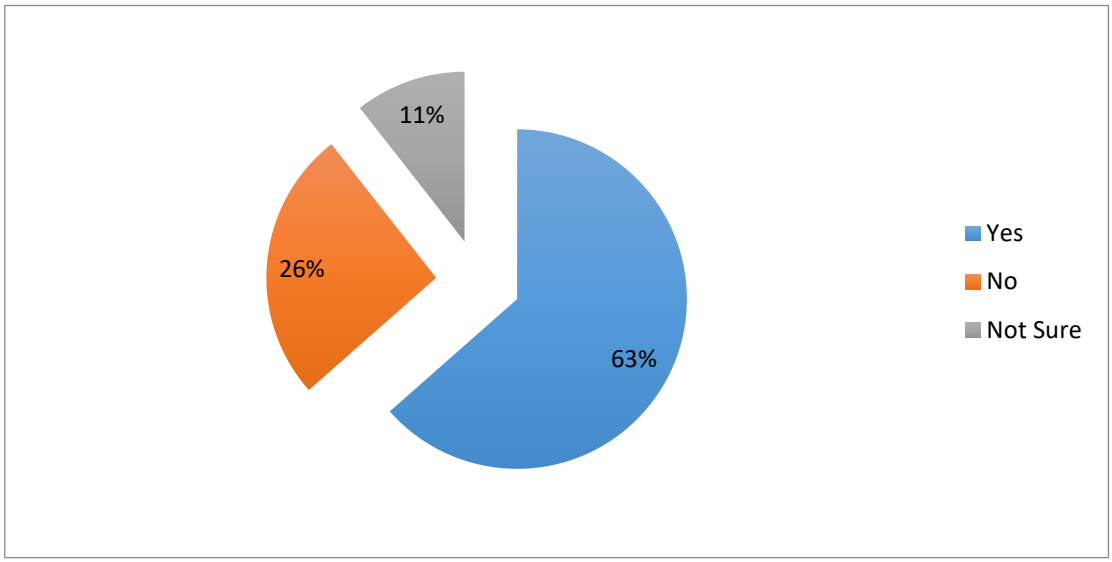


Figure 11 Tabulated & Graphical Results of Malware Handling Expertise

Analysis: The observation shows that a majority of the respondents (63%) had already handled a situation involving malware and considered themselves experienced in this area. As most of the respondents had experience in the malware handling process, the feedback regarding the following questions could be considered more informative and accurate.

Q-2	How many case studies related to malware do you receive for investigation?				
Options	< 5%	5%–20%	20%–50%	50%–80%	> 80%
189	29	20	40	60	40

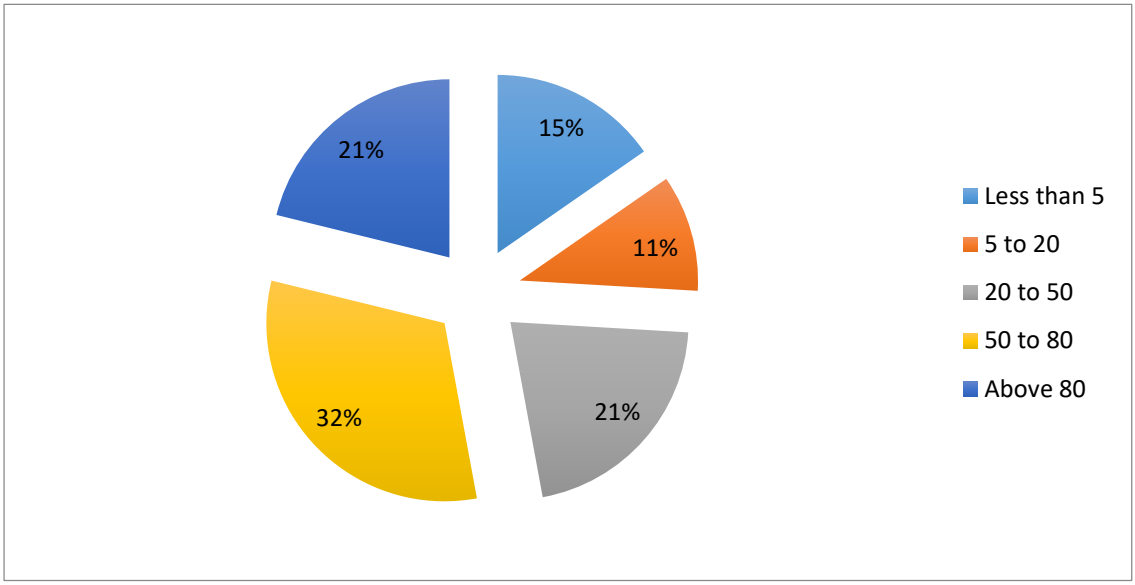


Figure 12 Tabulated & Graphical Results of Number of Malware Cases Received

Analysis: Most of the respondents (32%) received 50%–80% of the investigations into malware. A total of 21% of the respondents get up to 80% of the investigations related to malware. This shows that the majority of respondents (53% when adding the aforementioned percentages) receive at least 50% of their cases related to malware only.

Q-3	How many of the above cases were solved with accurate results?				
Options	< 5%	5%–20%	20%–50%	50%–80%	> 80%
189	10	40	79	50	10

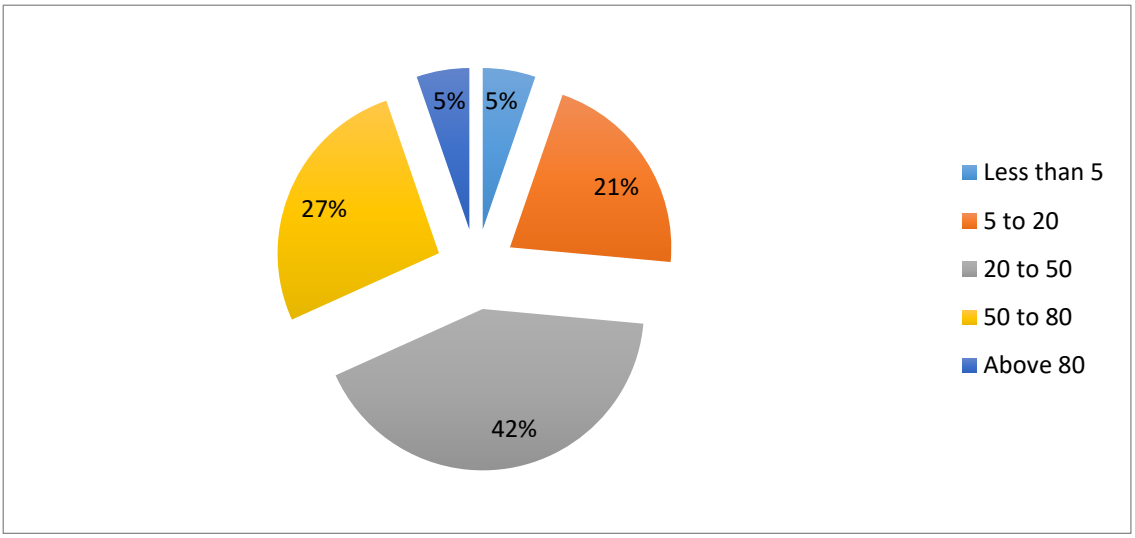


Figure 13 Tabulated & Graphical Results of Accuracy of Results Attained

Analysis: The above observation reflects the accuracy rate of detections with 42% of respondents' opinions suggesting that accuracy rates of 20%–50% were achieved. This result suggests that more than half of the cases involved inaccuracies or retained unsolved elements within the infected files. Higher rates of inaccuracy suggest that a gap exists between the actual findings of malware and the rate of infection; thus, there is possibly scope for improvement in the methodologies and implementations involved in malware detection.

Q-4	What is your opinion of the level of skills and technological tools adopted in your company to carry out the investigation process involving malware?		
Options	Weak	Average	Strong
189	30	109	50

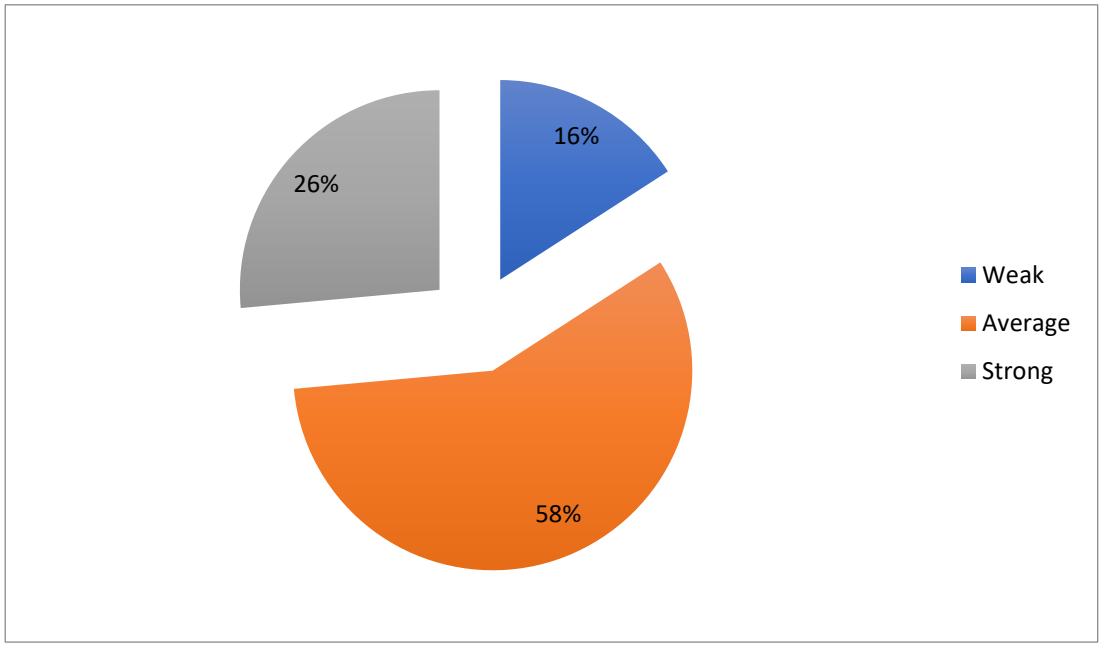


Figure 14 Tabulated & Graphical Results of Levels of Tools and Investigative Skills in Companies

Analysis: The level of skills and technological tools adopted for the investigation process was rated as 'average' by 58% of the respondents with only 26% rating the skills and tools as 'strong'. This suggests that the majority of users perceived the tools and the skills to use these tools for malware detection as ineffective, insufficient or in need of improvement to protect their systems from infections.

Q-5	What is the level of effectiveness in detection and response?		
Options	Weak	Average	Strong
189	60	90	39

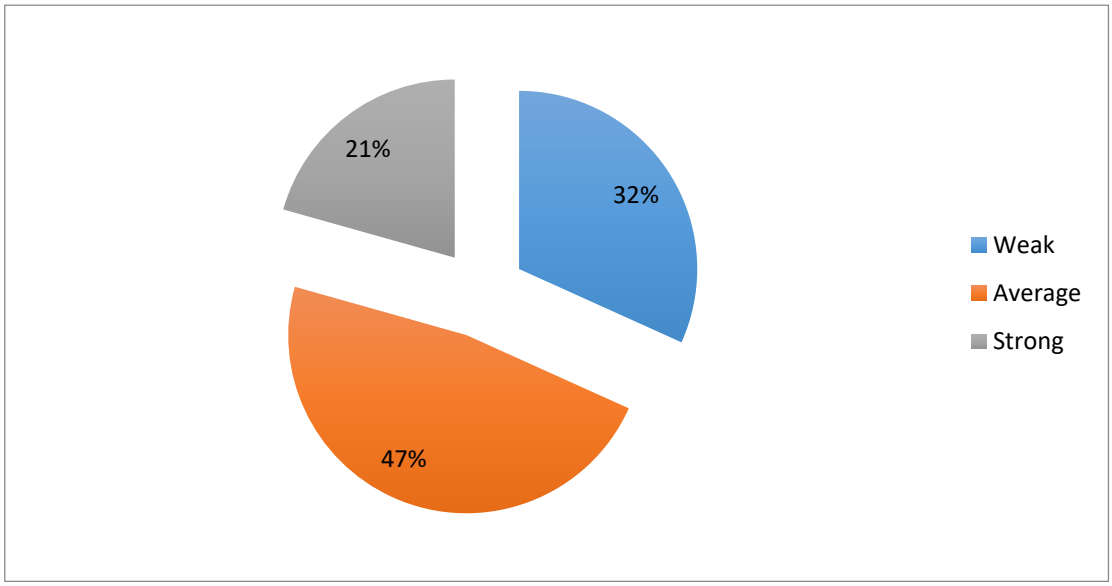


Figure 15 Tabulated & Graphical Results of Malware Effectiveness of Detection & Response

Analysis: These results suggest that the level of effectiveness in malware detection as perceived by the majority of the respondents (47%) is 'average'. Moreover, 32% perceived that the level of effectiveness was weak in terms of investigators and their performance. This result indicates that there is clearly scope to allow for improvements in detection and response effectiveness.

SECTION C: Questions based on malware detection tools and analysis

Q-1	**In your opinion, what kinds of tools have been adopted in malware detection and analysis?**			
Options	Open source tools	Shareware	Commercial tools	Cannot say
189	121	0	60	8

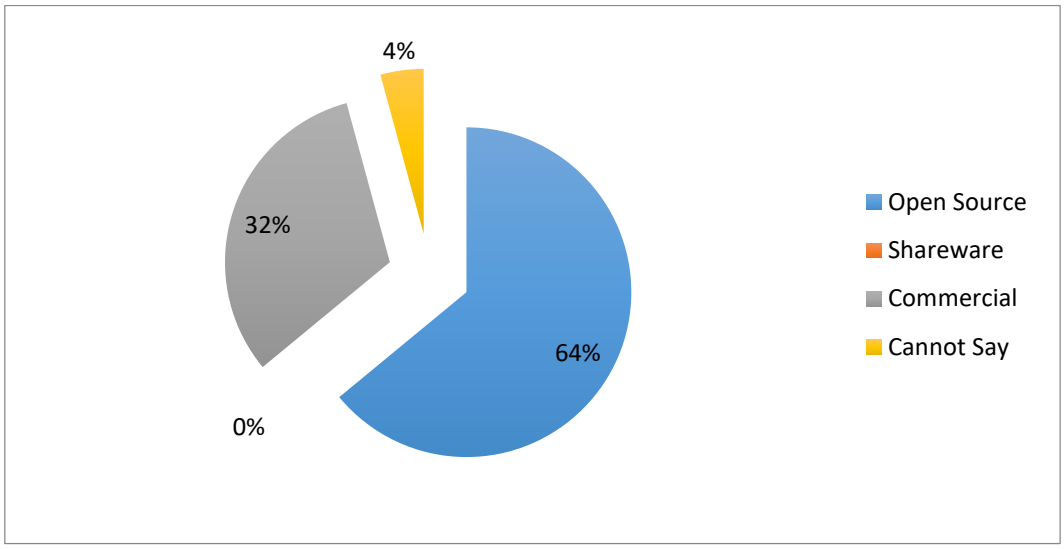


Figure 16 Tabulated & Graphical Results of Kinds of Malware Tools Adopted

Analysis: The results suggest that 32% of the respondents use commercial software, while 64% use open source tools. The common use of open source tools is an indicator of the acceptability and usability of open source tools when commercially available or shareware options are available. This result also follows the industry trend for use of malware detection software utilities commonly used, where the market split between commercial and open source tools is approximately equal (Vermesan & Friess, 2014).

Q-2	Do you think there is a need for a new customised tool for malware detection and analysis?		
Options	Yes	No	Cannot say
189	150	29	10

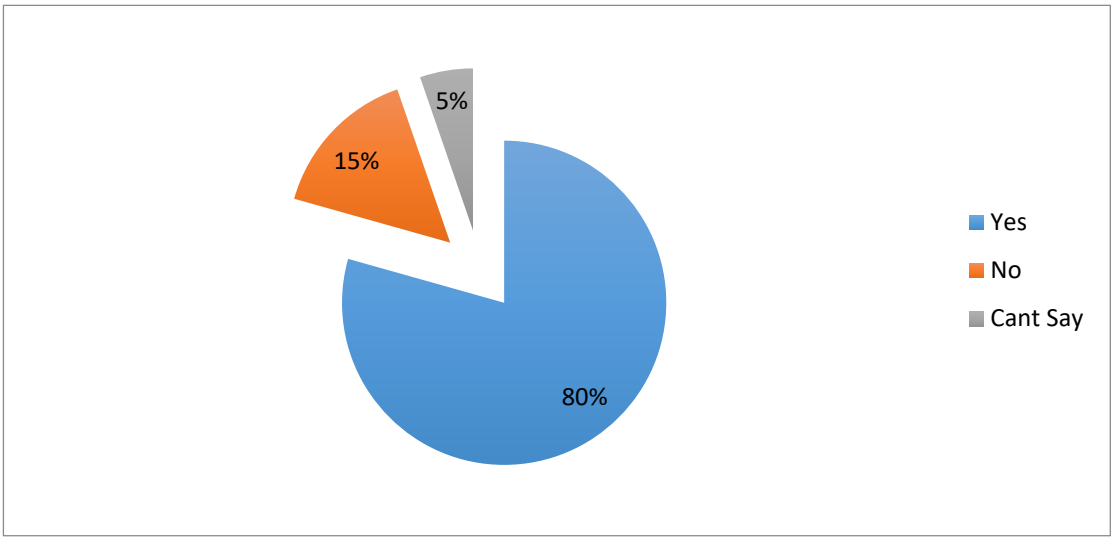


Figure 17 Tabulated & Graphical Results of Need of Customised Tool

Analysis: According to this observation, 80% of respondents perceive that there is a need for a new customised tool for malware detection. This result follows on from the previous questions relating to the opinion held that existing tool capability is considered poor in terms of effectiveness and a new customised tool is clearly required.

Q-3	Do you think cost plays an important role in adopting an individual analysis tool?		
Options	I agree	I do not agree	Cannot say
189	120	29	40

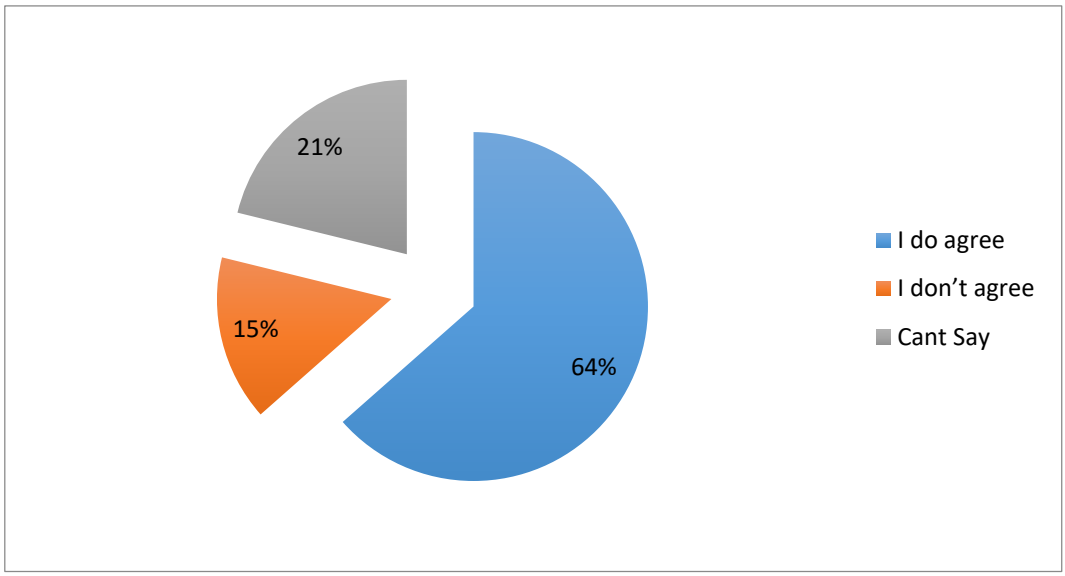


Figure 18 Tabulated & Graphical Results of Importance of Analysis Tool Costs

Analysis: According to the above observation, the majority of the respondents (64%) perceived that cost is an important factor in selecting and adopting an individual tool for malware detection. This result can play an important role in developing a framework at a later stage (mentioned in Section C, question 1) by considering the costs involved in any software tool to be developed.

Q-4	**In your opinion, which of the following methods is more viable and effective in analysing malware?**		
Options	Static analysis	Dynamic analysis	Other
189	60	109	20

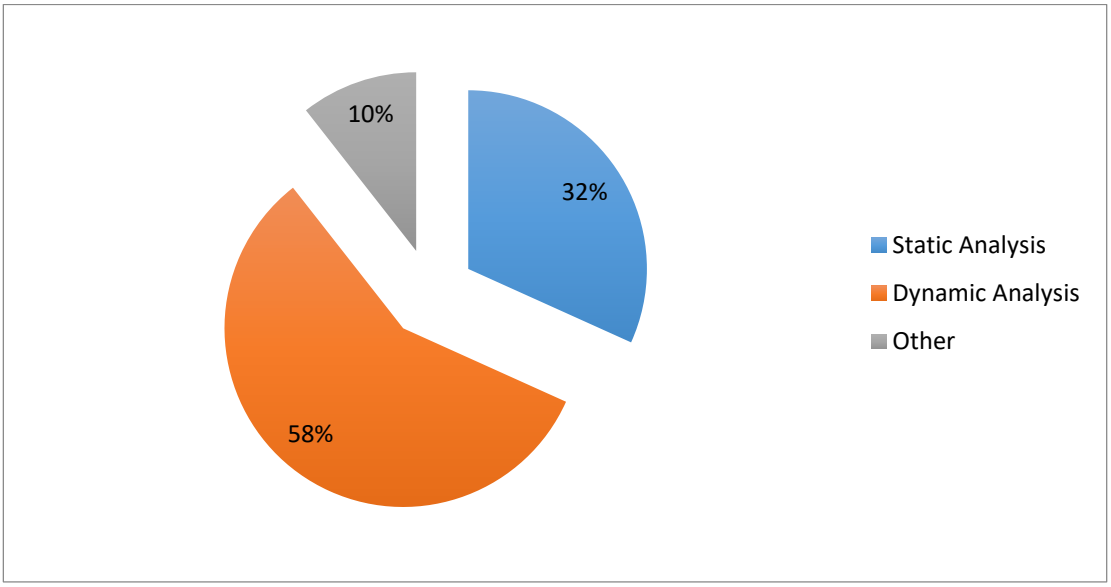


Figure 19 Tabulated & Graphical Results of Effective Methods of Malware Analysis

Analysis: Of the respondents, 58% perceived a 'dynamic analysis' methodology as the most effective method for detection, while 32% perceived a 'static analysis' method as most effective. This shows that there is no clarity in terms of whether a dynamic method is more viable and effective than a static analysis. The respondents' results here correlate with known advances in the development of malware detection methodologies where the latest tools are being developed using dynamic analysis methods as opposed to static (Aman, 2014), and dynamic tools are increasingly becoming commonplace.

Q-5	Is your tool equipped with a specific preloaded database for malware?		
Options	Yes	No	Cannot say
189	120	9	60

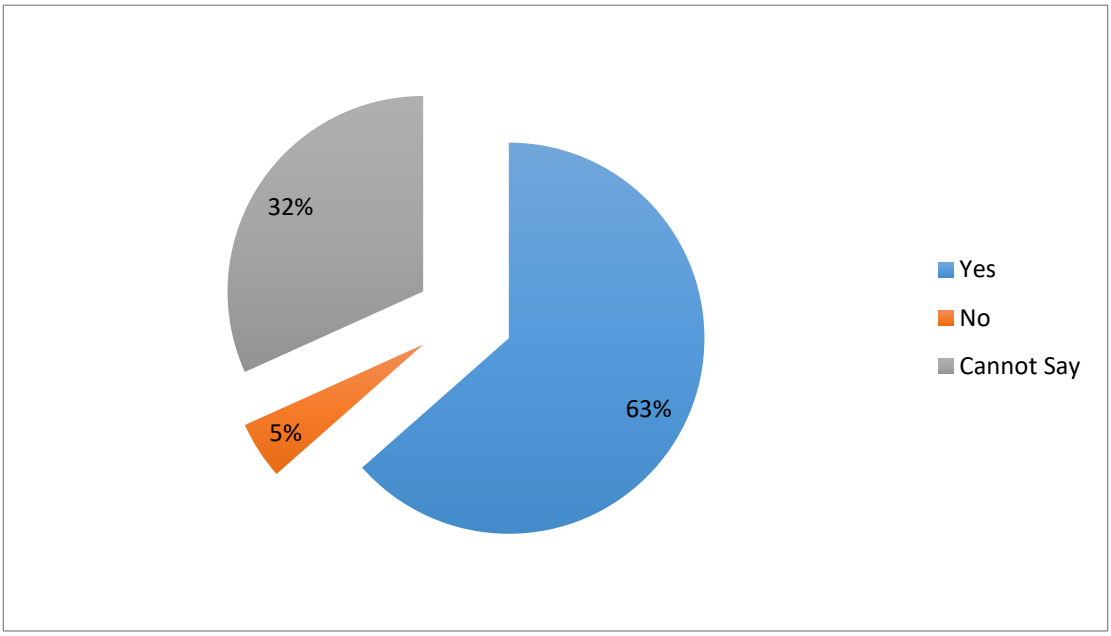


Figure 20 Tabulated & Graphical Results of Tools Equipped with a Preloaded Malware Database

Analysis: According to the above result, 63% of the respondents had tools equipped with a preloaded malware database while only 5% of the respondents were not using a preloaded database. This large number of preloaded database tools currently in use indicates that this is a necessary part of any tool and would be an expected feature within a detection tool. Therefore, it should most likely be implemented.

Q-6	If yes, how frequently do you update your malware database?				
Options	Every day	Weekly	Monthly	Annually	Do not do updates at all
189	110	55	14	10	0

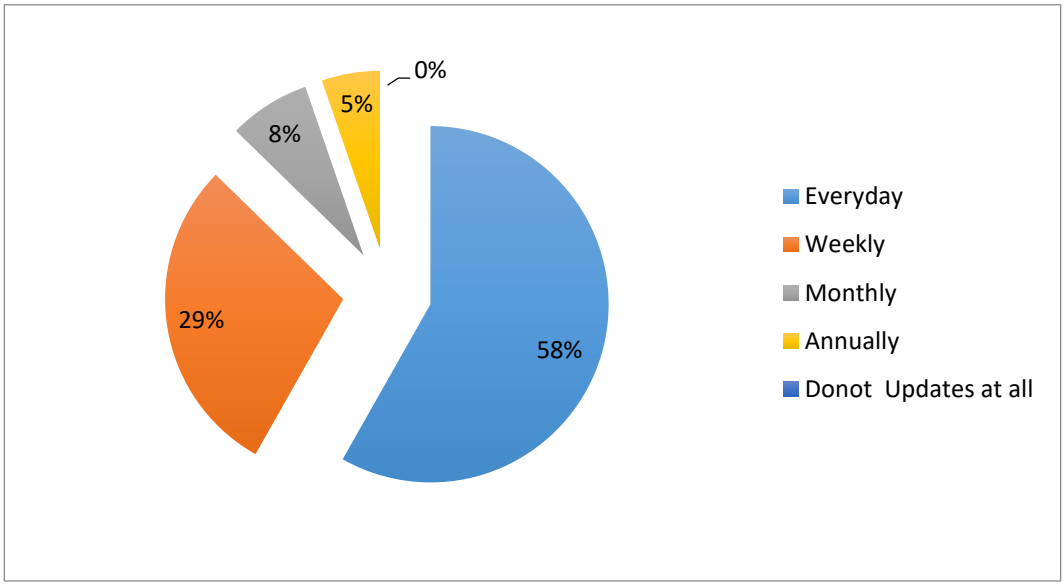


Figure 21 Tabulated & Graphical Results of Frequency of Malware Database Updates

Analysis: According to the above result, 58% of the respondents updated their malware database every day, while all of the respondents had updated the database at least annually. This shows that malware updates are an important part of the detection tool mechanism. This result is also reflected in the industry trend for malware databases to be updated automatically and as frequently as possible (Gibson, 2015) to coincide with the constant updates in malware itself.

SECTION D: Questions based on problems faced by investigators while conducting malware analysis

Q-1	**What is the core problem faced by the investigator during malware detection?**				
Options	Detecting the size	Analysing the behaviour	Detecting the level of infection	Proliferation	Database updating
189	5	40	80	44	20

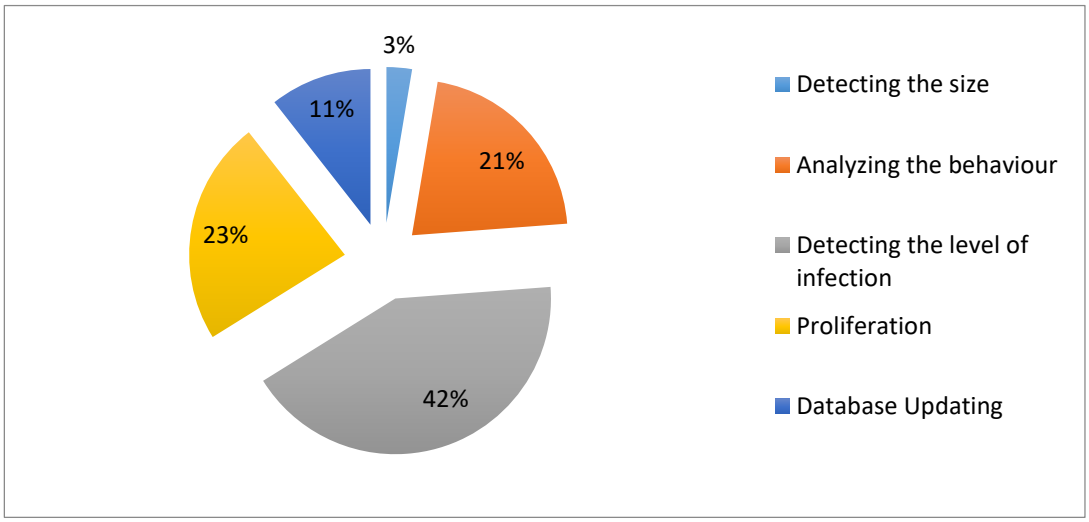


Figure 22 Tabulated & Graphical Results of Core Problems Faced by Investigators

Analysis: According to the result, 42% of the respondents perceived that *detecting the level of infection* is a critical issue. In addition, 23% perceived that proliferation was critical, and a similar number, 21%, perceived that analysing the behaviour of malware was critical. Although some problems were deemed more critical than others, it identifies that various problems are seen by experts and different problems need to be accounted for. Similar results can also be found for industry surveys looking at the ideal requirements sought in the

purchase of malware detection software (Intel Security, 2016), and no single function is considered adequate.

Q-2	What is the level of malware penetration in specific file processes?			
Options	Critical	Complex	Both complex and critical	Highly complex and critical
189	10	18	121	40

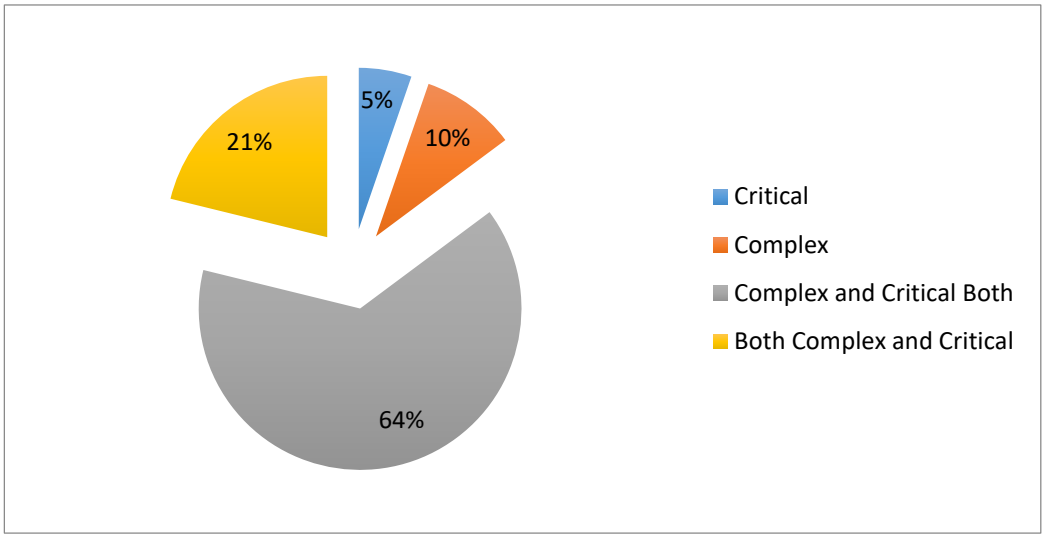


Figure 23 Tabulated & Graphical Results of Level of Malware Penetration

Analysis: Of the respondents, 64% perceived that the level of malware penetration is both critical and complex, thus indicating that detection is inherently difficult for an investigator. This result correlates well with published data on the increasing threat of difficult to identify (complex) malware, which are a high threat to system security (criticality) (Pollard & Mak, 2016).

Q-3	On average, how many days does it take for an investigator to detect the presence of malware in a file?				
Options	One week	Less than a month	Many months	Many years	Cannot say
189	70	55	50	0	14

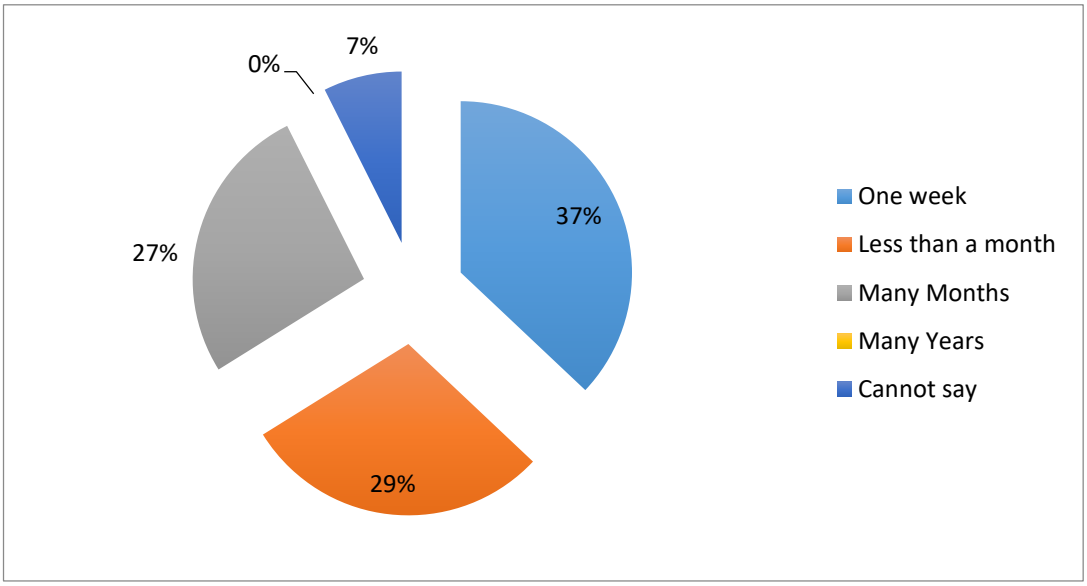


Figure 24 Tabulated & Graphical Results of Days to Detect Presence of Malware

Analysis: According to the above observation, 37% of the respondents perceive that an investigator needs less than a week on average to detect the presence of malware; with 29% of respondents perceiving less than a month. Although this information infers that the majority of current industry tools, under the use of experts, are successfully detecting malware presence in less than a month, the respondents were unable to give details on the method of the investigations, the tools implemented and the nature of the malware, thus making it difficult to draw any conclusions on the effectiveness of the tools themselves.

Q-4	What is an important parameter to consider during an investigation?			
Options	Malware signature	Malware behaviour	Level of infection	All of the above
189	19	10	10	150

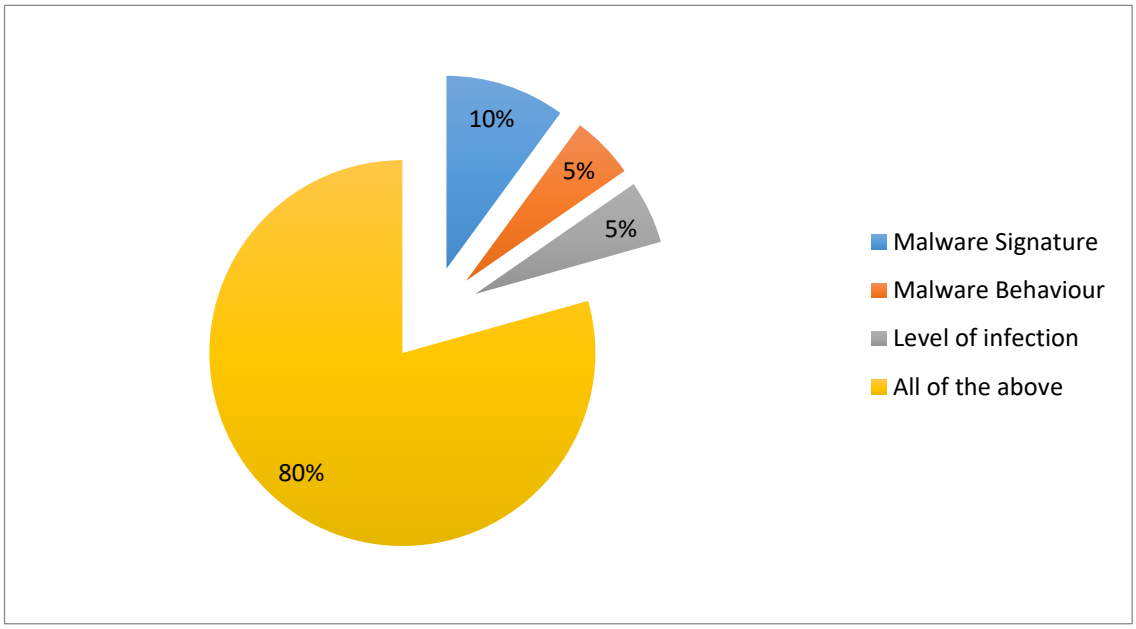


Figure 25 Tabulated & Graphical Results of Important Parameters during an Investigation

Analysis: Of the respondents, 80% perceived that it is essential for any investigator to consider all three of the following parameters: ‘malware signature’, ‘malware behaviour’ and ‘level of infection’. This demonstrates the need for a tool that performs all three activities to optimise the detection mechanism.

4.2.3 Critical Analysis

The results from the observations help to identify vulnerabilities in existing tools and methodologies used for malware detection. Although most of the experts were aware of the criticality and complexity of malware detection, no specific solution for accurate detection was mentioned.

It is noted that the sample size for the survey is small and, therefore, where applicable, literature data has been utilised in conjunction to validate the findings from the respondents. This has been used to help correlate the findings and instil more confidence in the survey results.

Selected parts of the analysis report are displayed in the following tables (Tables 2, 3 and 4).

Table 2 Important Factors from the Critical Analysis of the Questionnaire

Factors	Yes	No	Cannot say
Changes in malware landscape	50%	42%	8%
Organisational vulnerabilities	47%	21%	32%
Expertise on malware	67%	27%	6%
Detection of malware by non-technical employees	17%	53%	30%

Table 3: Level of Skill and Level of Accuracy in the Questionnaire

Level of Skill in Investigations		
Weak	**Average**	**Strong**
26%	58%	16%

Level of Accuracy during Investigations				
< 5%	**5%– 20%**	**20%– 50%**	**50%– 80%**	**Above 80%**
5%	21%	42%	27%	5%

Choice of Tools			
Open Source	**Shareware**	**Commercial**	**Cannot say**
48%	0%	37%	15%

Need for New Tools		
Yes	**No**	**Cannot say**
80%	15%	5%

Preferred Methods		
Dynamic	**Static**	**Other**
58%	32%	10%

Table 4: Important Factors from the Analysis of the Questionnaire

4.2.4 Further Analysis

Important factors from the analysis were identified to bring clarity surrounding the problem statement. The analysis resulted in the following insights:

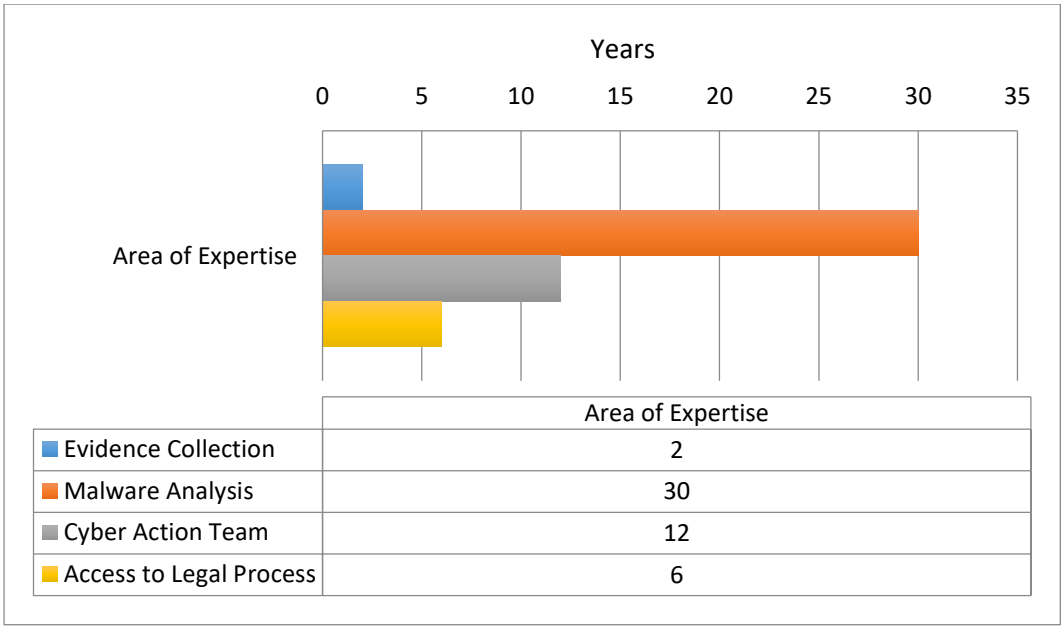
a) The core problem faced by investigators was to identify the level of malware infection in the file;
b) Most importantly, investigators relied on other parameters, such as the malware signature and combined behaviour, to make an effective detection;
c) Most of the infections were both critical and complex. Thus, there is a need for an effective method to reduce any associated complications;
d) Although there are tools that are capable of detecting malware and other malicious codes, most of them produce inaccurate or incomplete results;
e) The cost of the tool plays an important role in its adoption;
f) Most of the investigators wanted to use tools based on both dynamic and static methodologies;
g) It usually takes less time for investigators to detect the presence of malware but more time to investigate it forensically; and
h) Managing and updating the malware database is a task that should be given priority.

These points reflect the various challenges and problems that the respondents faced during their investigations and detections, effectively highlighting insufficiencies or misgivings within

the current methodologies and software usage. These results and analyses were considered when formulating a design solution to draft a framework and problem approach to produce an enhanced malware detection solution.

4.3 Market Research (Phase 2): Interview Analysis

This is an important phase of the research as it reflects the development of an artefact. Out of the 189 respondents from Phase 1, 50 respondents were selected to continue within this phase. The process of selecting candidates was initiated by scrutinising the respective profiles of all the respondents participating in the questionnaire process. The selection criteria are explained in Chapter 3 (Methodology) and the area of expertise of the selected respondents is shown below.



	Area of Expertise
Evidence Collection	2
Malware Analysis	30
Cyber Action Team	12
Access to Legal Process	6

Area of Expertise of Selected Respondents

4.3.1 Interview Analysis

The following analysis was done on the basis of the results received from the respondents. The analysis considers only answers given by the respondents; the author has included no personal opinions or recommendations in the analysis. The questions are as follows:

User Awareness of Malware Threats

According to the respondents, 64% are not aware of malware threats and their consequences. Various reasons for this were provided:

- A non-technical background;
- Being spoofed or bluffed, such as fake identifications;
- False temptations; and
- Lack of IT security.

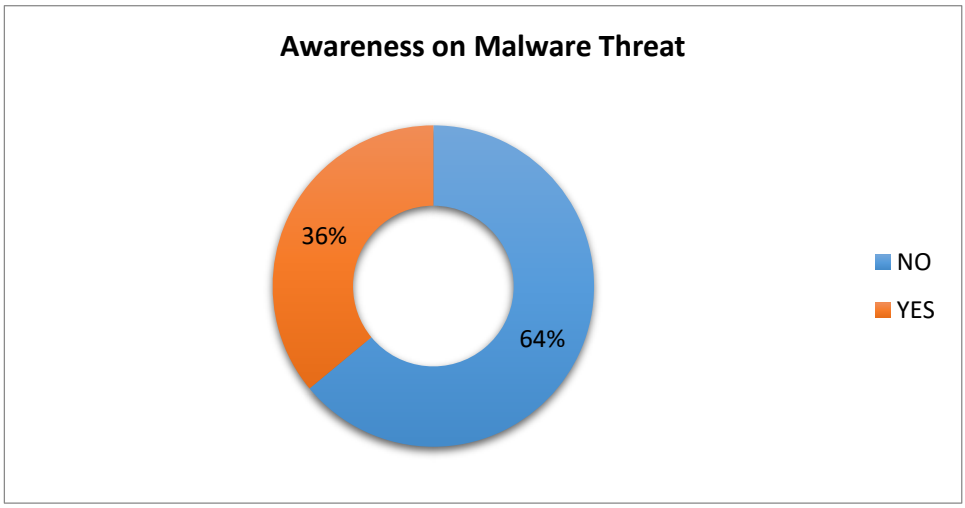


Figure 26 shows awareness of Malware Threats of Selected Respondents

Critical Analysis: The above reasons have raised concerns about the awareness of malware and related issues. In detecting malware, one of the most important aspects is the expertise in carrying out a malware investigation, which is vital for its success (Kaspersky 2017). This provides an important aspect to the research, proposing a user-friendly environment for detection, which enables a non-technical investigator to perform in a flexible and simple environment.

Difficulties in Malware Detection

When asked about the difficulties in detecting malware, 78% of the respondents agreed that they faced difficulties in malware detection, while 22% felt comfortable with detection. The most common reasons for difficulties in detection were as follows:

- The paucity of anti-malware products;
- Low accuracy rates;
- Lack of understanding of malware scripts;
- Lack of expertise in scripting languages;
- Poor reporting by the tool; and
- Loss of evidence.

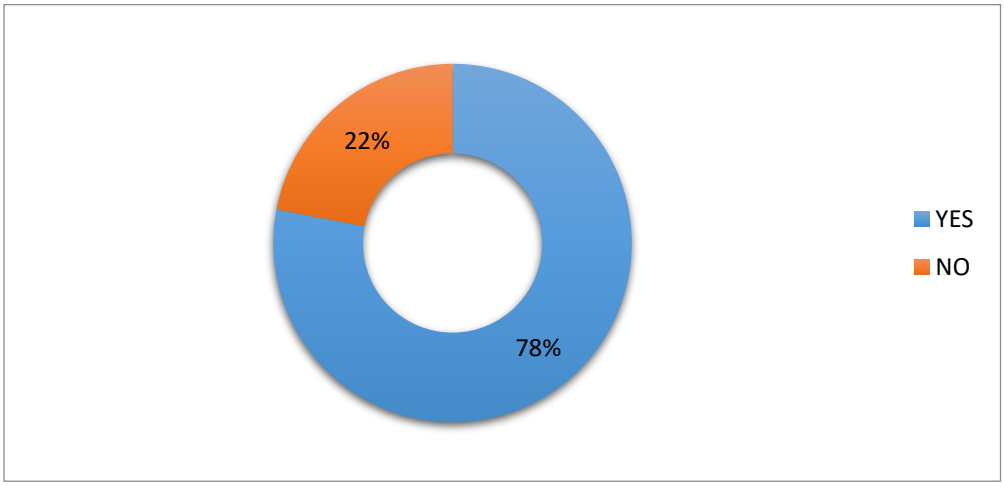


Figure 27 shows difficulty in Detecting Malware by Investigators

Importance of an Anti-Malware Tool

An anti-malware tool was deemed to be important because it could provide the following:

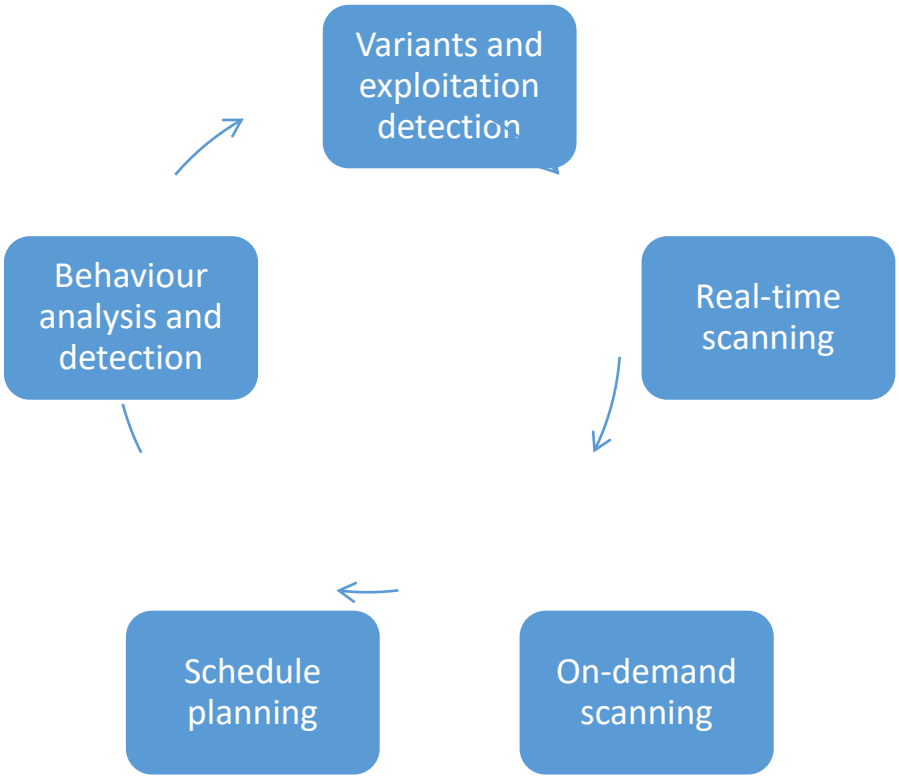


Figure 28 results of the Respondents for Importance of the Tool

What type of anti-malware tools would you prefer as an investigator?

Several recommendations and suggestions were recorded regarding anti-malware tools. Recommendations were categorised into three main types: ***open source***, ***shareware*** and ***commercial*.** Depending on the case study and the needs/requirements of the investigator, the most appropriate tool could be selected. Most of the respondents (72%) preferred open source as the recommended tool, thus making it an important result in selecting and developing the Toolkit artefact for the proposed solution.

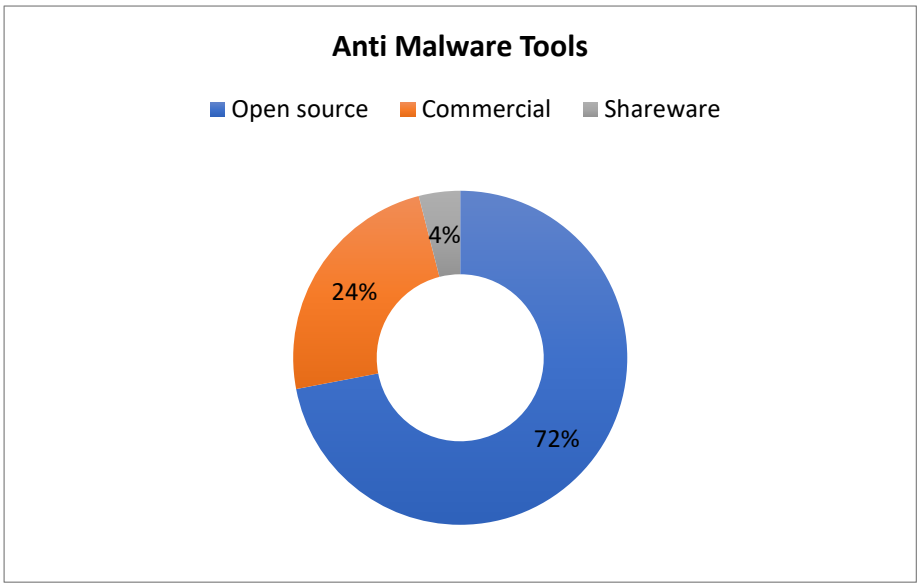


Figure 29 shows preferences for the 'Anti-Malware' Tool

Statistical Details on Malware Analysis

Observations on the types of malware affecting information security by investigators based on their personal investigative experience were recorded and converted into a statistical record. There were various common answers, Trojan horse being one of the most frequent.

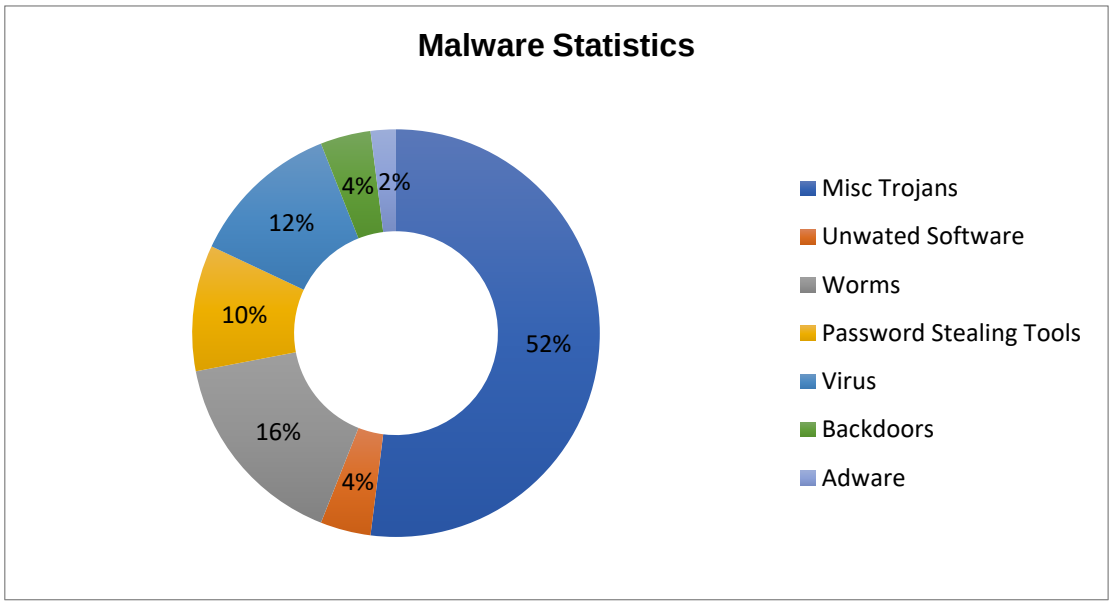


Figure 30 shows statistical Details on Malware Analysis

The statistical data (below) shows the existence of current malware and the types of online threats posed by malware

Online Threats by Malware

The analysis suggests that the malware domain needs to be considered when designing the malware database for the proposed artefact (Toolkit). The result below suggests that the maximum threat online is by denial of service (DOS) attack (46%), with the most probable source of an attack coming from a Trojan horse (concluded from previous chart).

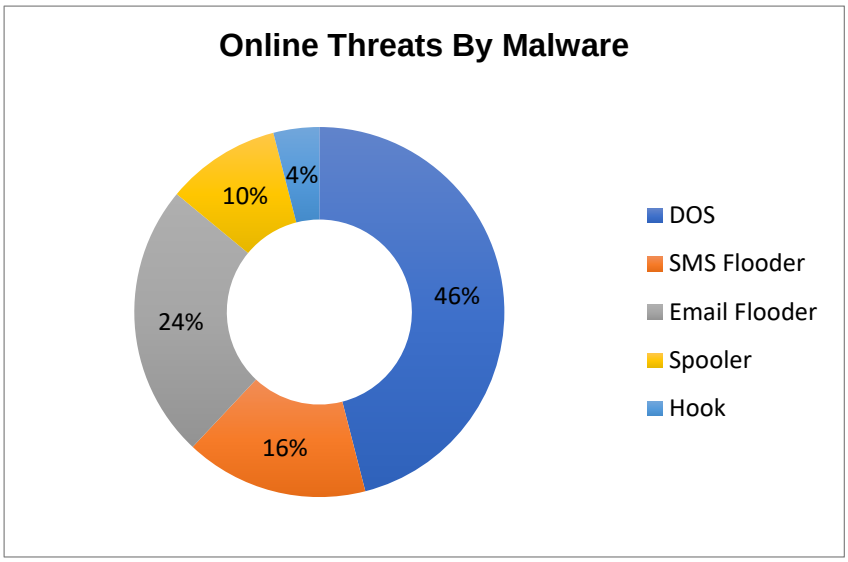


Figure 31 shows diagram shows the statistics for online Malware threats

Preferred Anti-Malware Tools Analysis Method

When asked about the preference for a static, dynamic, or a combination of both analysis tools, the combination of both was most preferred with 50% of the respondents giving this answer. However, some respondents still preferred the use of either static or dynamic methods. This result suggests that there is a requirement for both static and dynamic analysis tools. However, a combination of the two would be most preferential and is, therefore, sought after in a future developed tool. The results of the respondents' preferences of the analysis method are displayed below.

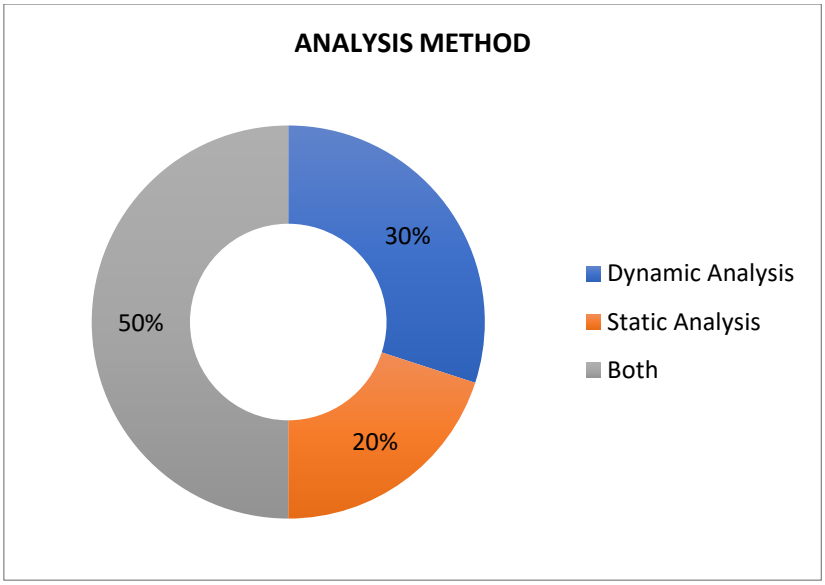


Figure 32 shows preferred Analysis Method of Respondents

Questions on the Tool's Performance Metrics

The questions on the findings illustrating performance are as follows:

Questions	**Yes**	**No**
Do you have difficulty with gathering information and then performing an analysis on it?	**65%**	**35%**
Do you think we need customised tools for malware detection?	**78%**	**22%**
Do you think we need any changes in the report structure?	**50%**	**50%**
Do you think collecting evidence is challenging?	**45%**	**55%**

Table 5: Results for the Tool's Performance Metrics in the Process of the Interviews

Tools Preferred by the Selected Respondents

When asked about preferences, the majority of the respondents (48) gave Wireshark as the most frequent answer for a tool to investigate behavioural analysis code. However, the respondents selected from a variety of tools and all were commonly used by at least 21 of the respondents.

As respondents were allowed to select multiple answers for any preference, the number of respondents has been used as opposed to the percentages. The results are displayed and tabulated in Table 6.

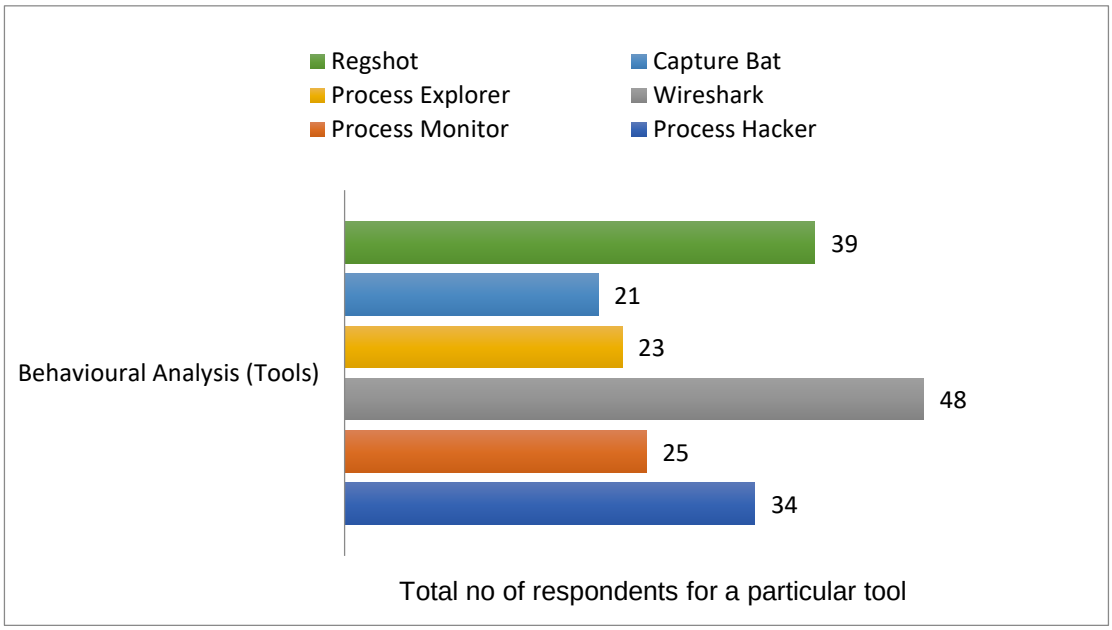


Figure 33shows preferred Behavioural Analysis Code of Selected Respondents

When asked about the preferred code analysis, the majority (36) of respondents selected IDA Pro as their choice. A similar number of respondents selected OllyDgb or LordPE as their preference, and a smaller number (11) selected Olley Dump. The results are shown in Table 6.

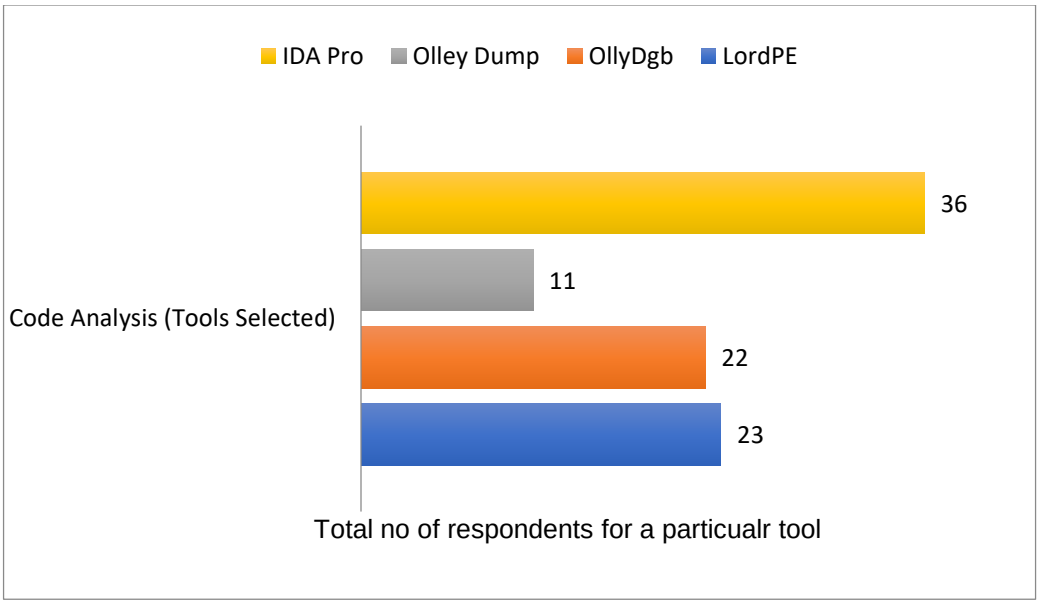


Figure 34 shows preferred Code Analysis of Selected Respondents

Process Explorer	Process Monitor	Wireshark	
Process Hacker	Capture BAT	RegShot	
Behavioural Analysis Tool			
LordPE	OllyDgb	**Code Analysis Tool**	
Olley Dump	IDA Pro		

Table 6: Results for the Respondents' Selected Tools for the Toolkit

Table 6 illustrates the combined results from the above figure as the total selection for both behavioural and code analysis tools.

Some similar patterns between the Interview Analysis and the Questionnaire

A comparison between the respondents' answers from the interview phase and the questionnaire phase was made to examine any similarities or differences between similar questions. It was noted that comparable percentages of respondents gave similar answers to the questions in common within each phase. The results of this comparison are shown in Table 7 below.

Questions in common	Questionnaire	Interview
Requirement of a new tool	80%	78%
Type of tool selected as open source	64%	72%
User awareness on malware and its detection	58%	64%

Table 7: Some Commonalities between the Questionnaire and Interview Analysis

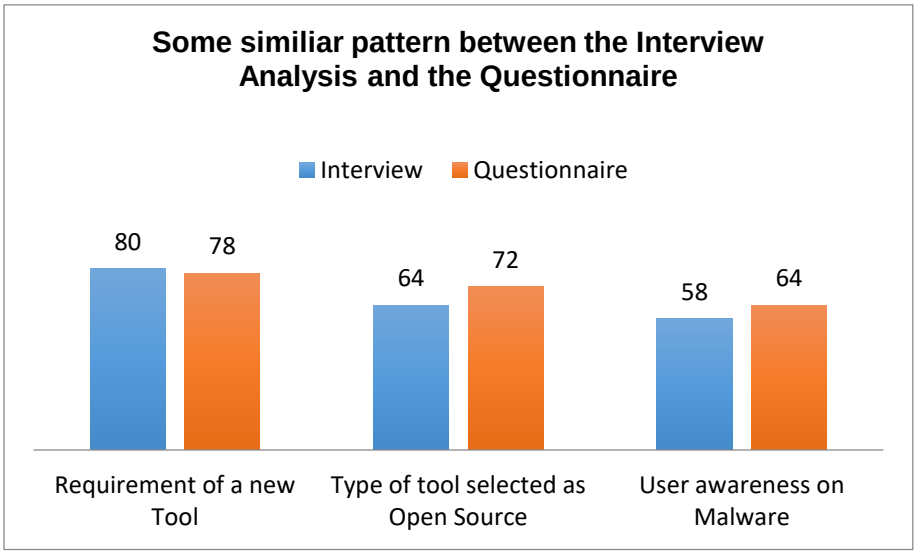


Figure 35 shows percentage Comparison Results between the Interview and the Questionnaire

4.4 Summary

This chapter discussed the implementation of methodologies and comprehensive market research, which provides the descriptions to quantify the basis of this thesis. Methodologies are the foundation of research and provide a direction and set of objectives for the author. The author used qualitative and quantitative methods, where questionnaire and interview analyses were employed to collect primary data. A market analysis acted as a guiding tool for understanding traditional methods of malware investigation, as well as helping to identify any ambiguities in their performance. The market research was divided into two phases: a questionnaire and interviews. Analysing the information gathered in the literature review (Chapter 2) helped to focus on important factors within the malware investigative industry. Limitations and concerns of existing tools from the perspective of industry-based users could thus be further explored through the questionnaire and interviews. The questionnaire was important for gathering and analysing a broad range of information to develop a possible new framework for malware detection and analysis by identifying any current tool limitations. The interview was conducted to obtain specific opinions on the challenges of existing malware investigation tools, which could be used to address the development of a new artefact. The analysis acted as a yardstick in this development and will be vital in future chapters.

The research was conducted by selecting 189 respondents who were categorised by professional position and divided into two separate research phases.

In the first phase of the research, selective questions related to malware, detection techniques and challenges were asked to precisely determine the current needs and requirements of the investigators. The results focused on the problems faced by investigators and discussed the

vulnerabilities in detection. They also illustrated the need for using both a static and dynamic analysis approach in malware detection and for maintaining enhanced costs and resources.

The sample size for this first phase could be considered small. However, due to practical considerations of locating appropriate respondents, the similarities in the nature of the results across different locations and roles, and with consideration of the research objectives, it was deemed sufficient for this study.

In the second phase of the research, through interview analysis, findings were discussed in terms of the respondents' area of expertise, user awareness of malware threats, knowledge of difficulties in malware detection, the importance of anti-malware tools, the preferred analysis method and preferred analysis tool.

For the second phase, a relatively small sample of 50 respondents was selected from the original group of respondents based on selected criteria. Although the sample size was further reduced, it was important to be selective regarding the participants' roles and experience within the original sample of respondents for the interviews. As this second phase consisted of finding the opinions of professionals in the correct field of expertise, collecting information from an appropriately experienced sample was considered a higher priority than a large sample with less experience. It was therefore considered justifiable to use a smaller sample size for the objectives of this research.

Similarities were found between the questionnaire (phase 1) and the interview (phase 2) with clear correlations in percentage results for difficulty in investigation, requirements of static and dynamic methods, the need for a new framework and satisfaction with existing methods. The closeness of the results from the two phase instils confidence in the validity of the overall responses gathered from the market research.

The results gained from the market research with the IT experts in malware tie in well with the findings from the literature review into malware as a whole. The majority of the respondents were aware that malware is a growth area in which the level of threat is increasing, something that is well documented in statistical data from the literature review. The respondents identified that the effectiveness of malware detection could be improved with almost 80% rating their current systems as average or weak. This was shown to be critical in selecting appropriate tools through the literature review where effectiveness and efficiency were key drivers, suggesting that available tools were not meeting expectations. The challenges associated with malware detection were discussed within the literature review, and the respondents were also aware of these challenges with the majority recognising the need for technical employees within the field, the necessity of frequent updates to stay on top of current malware trends, and the reality that malware threats are complex and critical in their penetration. Finally, the respondents could see the need for both static and dynamic analysis methods with some preferring one method over the other, and the benefits of both were highlighted in the literature review.

The results from both phases of the market research enabled the author to develop needs and requirement criteria for the artefact, covering all necessary steps in its development (Chapter 6).

The following is a summary of user needs, which will be used for developing a proposed framework:

- Solution to identify the level of malware infection in the file;
- Develop a malware detection and analysis framework;
- An effective method to reduce any associated investigative complications;
- Enhancing current accuracy and efficiency;
- Developing a cost-effective toolkit;
- Framework based on both dynamic and static methodologies;
- Framework that can detect the presence of malware in an effective timeframe;

- Managing and updating the malware database is a task that should be given priority; and
- Maximum utilisation of open source tools rather than commercial ones;

CHAPTER 5 - A FRAMEWORK FOR EFFECTIVE MALWARE DETECTION AND ANALYSIS

5.1 Overview

The current chapter proposes and describes the rationale behind an effective malware detection and analysis framework. In the literature review (Chapter 2), various studies (Ayers, 2009; Richard & Roussev, 2006) identified vulnerabilities and limitations in the existing forensics tools for investigating malware, and this statement was still found to be true according to the results obtained in Chapter 4, where data analysis demonstrated some common flaws in the existing investigative process for detecting malware in information systems. Thus, there is a need to develop and deploy an effective system that assists investigators in the accurate and complete detection and analysis of malware without compromising data integrity. The market and questionnaire analysis in Chapter 4 showed a need for an extensive system that can handle both static and dynamic investigative methods. The analysis also demonstrated that current software detection tools are insufficient because they may produce inaccurate and incomplete results. As demonstrated through observations and research, there is sufficient room for the establishment of a new framework so that investigators can employ efficient malware detection techniques that are effective at minimal cost.

Based on this, the subsequent chapter describes a new approach called the triple-tier centralised online real-time environment (tri-CORE) malware analysis (TCMA) method, which was developed as part of the current research and, as such, is considered a contribution to the field. Current detection methods and their associated issues and challenges are also outlined to justify the TCMA method and to show how it addresses the challenges.

5.2 Different Approaches to Detection and Analysis

In digital forensics, methods are carried out based on algorithms, which exist to accomplish process objectives. For the detection and investigation of malware, every procedure includes two systematic functions: the malware detector (MD) and the source code (S). The specific domain of the MD depends on the domain of the S, which is defined by the mathematical 'set' as malicious or genial (Neelakantanand & Rao, 2008).

The process is initiated through the MD function, where the MD scans the S to identify where it falls on the 'set' (malicious or genial). The MD presents the results with the matching technique to check stored signatures in the database. The investigation may produce a distinctive result as a false positive, false negative or hit ratio (Neelakantanand & Rao, 2008).

A false positive result appears when the system detects malware in a non-active (uninfected) database. The attributes of the detected malware are unique, allowing for easy detection in a non-malicious file. A false negative result occurs when a detector is unable to find the malicious source code in an infected database. This situation usually arises when a scanner or detector fails to match the signature or lacks the sample of a signature, making the scanner vulnerable. In contrast, the hit ratio is the expected output that all investigators want to achieve. The hit ratio only appears when a detector successfully detects a malicious code accurately and when the possible signature matches the attributes of the infected file.

These abovementioned procedures were considered as the fundamentals for proposing a new framework, which provides a definitive pattern for the existing study. Before proposing a new framework for malware detection and analysis, it was critical to perform a comprehensive study in this chapter on the existing methods and to study their associated issues and challenges. This provided a clear picture of the existing working model and its associated deficiencies and vulnerabilities, which were later acknowledged by our proposed system.

Although there are various categories to differentiate malware detection methods, three distinctive methods are popular among investigators: 1) the signature-based method; 2) the specification-based method; and 3) the behaviour-based detection method (Demme et al., 2013; Egele et al., 2015; Vinod & Laxmi, 2009; Volunkin & Skormin, 2006; Xu & Sung, 2008). Each method poses various issues and challenges.

5.2.1 Signature-Based Detection and Analysis

Signatures result from the concatenation of bytes, which is an integral part of malware or malicious codes. Malware can be either polymorphic or metamorphic. Polymorphic malware is more common; here, the identity and source of the generations are usually hidden, making it difficult for investigators to collect evidence against the plotter (Demme et al., 2013). In simple malware, the source code, or the initial point of reference of the infected program, is reformulated and complete control of the system is transferred to the malicious payload. Each source code mutates, but the original system data remain intact. This type of malicious program uses a model-based engine that replicates malware viruses each time the user runs the program on the system. Metamorphic viruses re-program the source code by modifying the basic attributes of the parent program and producing a completely new signature for each parent–child variant created in later stages. Thus, due to the nature of metamorphic or polymorphic viruses, it is difficult to trust signatures as their integrity will always be at risk. Therefore, through the research paradigm, the researcher has ruled out this approach and has developed a robust solution (database sets for every modified signature; mechanisms such as entropy were included) to tackle the abovementioned issues to enhance the signature detection mechanism.

Associated Problems and Challenges with Signature-Based Detection and Analysis

The issues related to signature-based detection and analysis are as follows:

a) Extracting and distributing the signatures for investigative purposes is complex and time-consuming;
b) A distinctive infrastructure for the investigator to perform testing is required, such as using penetration research and analysis, a manual process conducted in a controlled environment;
c) The signature from an infected file is often bypassed during investigation, requiring periodic updates for new signatures before the time of detection and the analysis process; and
d) As new signatures are added, the database and malware signature repositories become too large to store practically or to manage optimally.

5.2.2 Specification-Based Detection and Analysis

Specification-based detection and analysis depends on the details of the source code implementation and the deployment of a particular system file or application. The process requires thorough knowledge and expertise of each application development phase. This process also adopts a hypothetical approach, assuming that each abnormality is proof of a malware carrier. The process uses concepts of reverse engineering, which focus on reaching the source or initiator during the investigation. Sometimes, multiple run-ups are required to debug the source program.

Problems and Challenges Associated with Specification-Based Detection and Analysis

The main issue with this technique is that it requires complete accuracy and knowledge of system application development. The whole process is performed using predefined procedures that must be followed by an investigator, making it difficult to carry out in a real-time

environment. Moreover, the system's procedures must be updated on a regular basis, creating new challenges for investigators.

5.2.3 Behaviour-Based Detection and Analysis

Most malware detection systems perform automatic detection based on observations of anomalous behaviour and system activity. Specific pieces of the actions and reactions created by malware harm the system with a specific purpose. One method of analysis is to study the sequence of the calling program through the operating system. These application program interface (API) activities, which use parameters to identify suspicious activity by eliminating and dismissing harmful processes running on the system, can be intercepted by behaviour-based malware detection software. Experts have attempted to create various parameters to analyse the self-replicated, repeated patterns in infected areas

Problems and Challenges of Behaviour-Based Detection and Analysis

The problems related to behaviour-based detection and analysis are as follows:

a) Behaviour-based detection and analysis focuses on selecting files in the system randomly, which can allow malicious content to avoid detection. Even if the malware is detected at a later stage, its infectious consequences cannot be reversed, creating a detection failure (Volynkin & Skormin, 2006); and

b) The method is inappropriate for recording previously unknown malware information. This allows for malware proliferation, making the system unreliable and the investigation complex (Egele et al., 2015).

All of the aforementioned analyses depend on continuous monitoring processes while the suspected programs are running. These methods are very risky, as infection could cause harm and loss before the system identifies a program as malicious (Xu & Sung, 2008).

5.3 Evaluation of Existing Malware Detection Frameworks

Evaluating various existing frameworks rationalises the need for a new proposed detection method. Table 8 depicts the criteria, explaining the limitations of the outlined methods with a view to justifying the requirement for the new method. For evaluation, the selection of parameters were based on a literature review and analysis (Chapter 2), and distinguishing features were based on accuracy, performance and functional operations. The new framework would relate to a completely new viewpoint, while acknowledging all previous limitations mentioned in existing frameworks. This analysis was critical in proposing the solution as a framework for the given problem statement. Special consideration was given to these limitations and was proposed as a critical feature in the new system.

Parameter	**Limitation of Current Framework**
Recovery	Only a few anti-malware tools (Spybot, Norton, etc.) provide auto-recovery functions, but the ones that do still fail to detect residual traces of malicious codes in infected files (Sharif et al., 2008).
Usability	Most of the frameworks are designed to detect only processes that are untrusted and suspicious and non-functioning malware remains active (Reiter & Yen, 2008).
Performance	Existing methods lack robust monitoring systems to analyse the existing stage of the central processing unit (CPU). A higher workload during analysis leads to a higher chance of downtime (Yin, 2007).
Functional Operations	The framework proposes either static or dynamic functions, and both have their own limitations; still, no system carries a combination of both (Kolbitsch et al., 2010).
Reinfection Detection	A process of reinfection is very common, but there is no special consideration of the reinfection phase in most existing methods (Sharif et al., 2008).

Table 8: Limitations of Existing Frameworks using Selective Parameters

Detection Method	Accuracy	Performance	Functional Operations
Signature-based	The likelihood of not understanding malicious behaviour is high; thus, there is no accurate or appropriate method to represent malware with a correct signature every time (Barabas et al., 2013).	As this method is very dependent on human expertise to generate a signature, the likelihood of human error is greater, resulting in fluctuating performance (Li et al., 2008).	The process is time-consuming, as an expert is needed to add new signatures each time a malware infection is detected (Newsome et al., 2005).
Specification-based	A major limitation of this method is the level of inaccuracy of the results. It is difficult to maintain accuracy when analysing the behaviour of malware based on pre-defined specifications (Fortinet, 2017; Moser & Kruegel, 2007).	This method is based on attaining a fixed result of a certain set of rules and patterns. It is a complex method, as explaining the total behaviour of malware through fixed rules is difficult. Thus, the chances of detecting invalid behaviours become more likely (Moser & Kruegel, 2007).	This method uses system calls to understand the malware specification, so issues such as detecting malware shadow processes become infeasible, thereby limiting functionality for the investigator (Ma et al., 2010).
Behaviour-based	This method is based on comparing recorded behaviour with detected malware. If the exception in the detection phase has never been recorded, then the chances of obtaining an error increase. Thus, the method is prone to false detection (Brumley et al., 2007).	If analysis of the infected files is performed in an inappropriate environment, the chances of inappropriate results increase. Hence, this method does not enhance overall system performance (Liu et al., 2007).	This system lacks major functional attributes. For example, when malware infection causes a loss of data confidentiality, this cannot be detected and can result in limited operations on the client's machine (Brumley et al., 2007).

Table 9: The Distinctions between Current Malware Detection Methods Based on Parameters

5.4 Requirements for the New Framework

Table 8 gives a comprehensive analysis of the limitations existing in the current malware detection frameworks and methods. These limitations and vulnerabilities attract a significant increase in malware attacks and exposure, which results in the loss of the information security triad (CIA); that is, *confidentiality*, which is detrimental to the *integrity* and *availability* of data (in Barabas et al., 2013; Li et al., 2008; Ma et al., 2010; Moser et al., 2007; Newsome et al., 2005). Current malware detection mechanisms and methodologies suffer from inefficiency and inaccuracy as they concentrate on specific malware categories, which can be easily negotiated through obfuscation techniques. There is a great need to research and focus on a new formalised approach to collect, analyse and remove malicious content from a system. To address the challenges faced by investigators, there is possibly a need and, therefore, an opportunity to use tools based on both static and dynamic techniques (hybrid technique), which are able to address associated problems with malware detection. Duo combination has been used before in various frameworks for investigation, but due to limitations and challenges associated with the speed and timeframe, it has had limited success in the field of malware investigation. However, user feedback (from Chapter 4) regarding slowness was not an apparent result in a succession of artefact operations, thus was widely accepted for our proposed framework. Moreover, in the proposed framework, the duo was integrated only after the acquisition of malware, which makes the solution more effective for detection and analysis. Having this functionality can assist the system in analysing the malware sample before starting an investigation. In a situation where the attributes of a sample infected file or a captured malware match the attributes in the database, it directly jumps to the solution. This makes the

system more efficient in terms of time and cost. Moreover, these two methods were kept tightly coupled, more agile and comprehensive in nature to provide better results.

Based on the above challenges and conditions faced by the investigator in detecting and analysing malware, it is essential to propose a new effective framework to enhance the ability to obtain results. Through rigorous research and expertise, the researcher has identified the issues and made a new model, which appears more effective and efficient in terms of usability, performance and flexibility. To ensure that the analysis is comprehensive, the framework works with both manual and automated processes in detecting and warning the system. The aim is for the proposed framework to become integrated with the present system to detect and analyse malware activities. To understand the nature of the necessary changes, an extensive search of existing market availability was conducted and analysed in the field to gather information to develop a new framework. The results, which were outlined in Chapter 4, illustrated problems faced by investigators and the vulnerabilities that exist in malware detection. The following are some of the important insights from the market analysis, which were identified to bring the problem statement into clear view to propose new framework:

a) The core problem faced by investigators was to identify the level of malware infection in the file;
b) Most of the infections were both critical and complex. Thus, there is a need for an effective method to reduce any associated complications;
c) Although there are tools that are capable of detecting malware and other malicious codes, many produce inaccurate or incomplete results;
d) Most of the investigators wanted to use tools based on both dynamic and static methodologies;
e) Managing and updating the malware database is a task that should be given priority;

f) Open source tools were favoured over commercial or shareware-based tools during malware detection; and

g) There is a need for more customised tools for malware detection.

5.5 The Proposed Framework: The TCMA Method

The required framework was designed as an effective malware detection and analysis tool based on the results obtained from the secondary research conducted for the survey report. The proposed framework is called the triple-tier centralised online real-time environment (tri-CORE) malware analysis (TCMA) method. The method includes three distinct phases of detection and analysis in which the entire research pattern is divided into three different domains called tiers. The tiers can be defined as the malware acquisition function, detection and analysis, and the database operational function. Using a technique of encryption in the model to improve the security level of the code itself from a malware attack, these tiers help ensure the complete integrity and availability of required information during the investigation and also help provide a comprehensive approach to ensuring document fidelity. Defining tiers is a completely novel approach developed by the author to make the model more effective and efficient. This approach provides an independent process layer, which links various independent malware detection approaches towards a single end-to-end execution system, removing all the associated static constraints in the study. The TCMA method defines various policies, categorises organisational assets and creates a new environmental variable for the process. The framework is designed to confirm malware infection using any standard of deployment. It is an unbiased model that initialises its operation by first confirming the malware infection and then by analysing, detecting and proliferating it. The objectives in developing this framework were to understand the detection costs associated with dealing with malware discovery and analysis within an associated environment, and to provide accurate and precise information to enhance the performance matrices. As mentioned above, the entire process is divided into the following

three phases or tiers:

a) The malware acquisition function;
b) Detection and analysis; and
c) The database operational function

Figure 36 defines the relationships between these different phases of the process. Each phase is explained with consideration of its implementation.

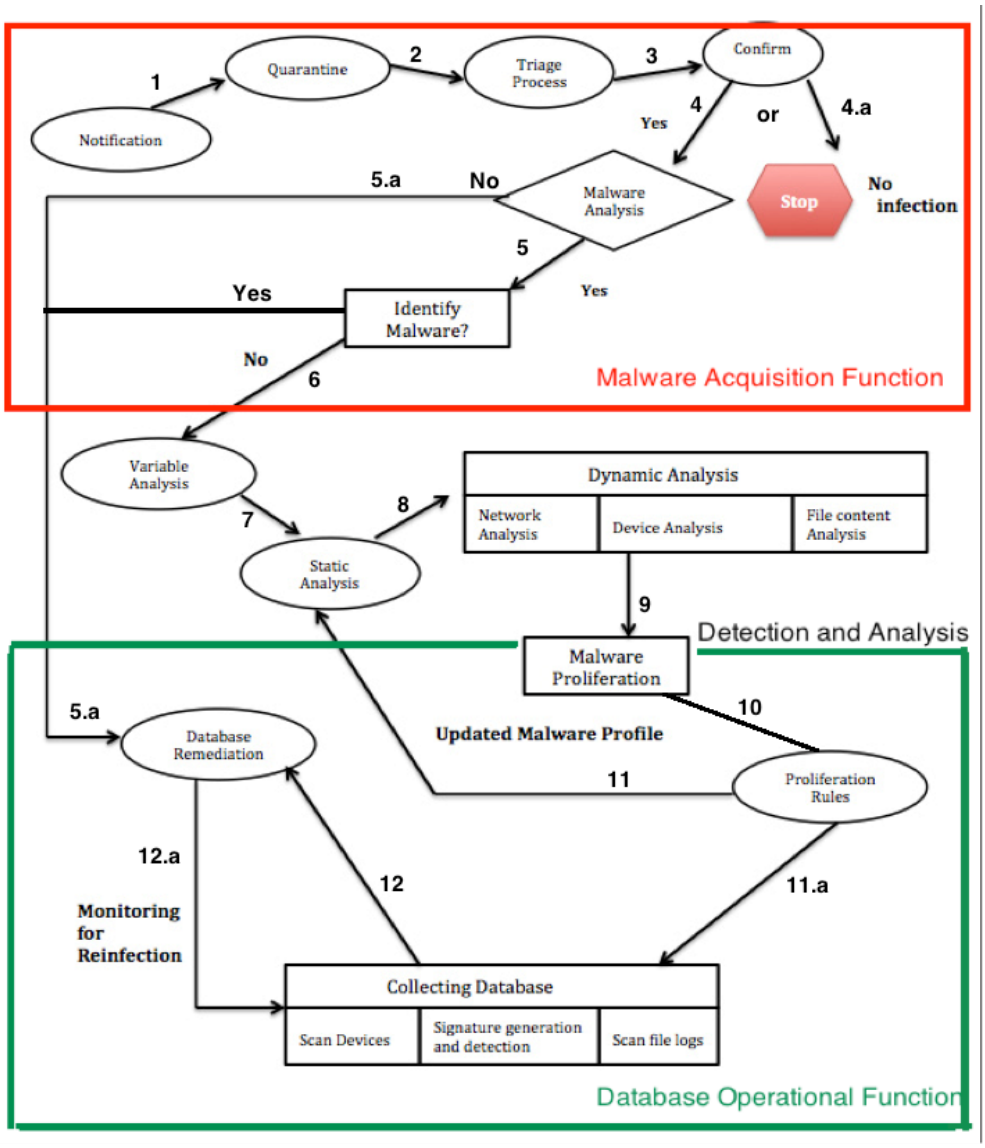


Figure 36 Illustration of the Three-Phase Process for TCMA (Source: author)

5.5.1 Phase 1: The Malware Acquisition Function

Phase 1 begins by identifying the presence of malware code in the suspicious file. It deals with the operations and processes concerning the confirmation that a particular file is infected by malware. In Phase 1, the author has added functionalities (mentioned below) that can be used to detect both active and passive infections in a compromised system. This phase includes the

following components:

a) Notification: Acquisition can be initiated in a number of ways, including alerts from a third party (cash management system [CMS] or any payment gateway system or intrusion detection systems [IDSs]) or from a destination suite. This function acts as a checkpoint to determine whether the issue occurring in the system is realistically a threat (Park, 2012);

b) Quarantine: The main objective of this functionality is to collect the infected files from the system to separate them from the rest of the healthy files on the system and ultimately remove them. This prevents any reinfection or replication of malicious code from being transmitted to other files (Park, 2012);

c) Triage process: This function, also called the *identifier*, scrutinises the criticality present in the malware infection. It contains a process of analysis to categorise the level or status of the problem (high/medium/low degree of seriousness; Blount et al., 2016); and

d) Infection confirmation: Before this stage, investigators have obtained enough information on the status of the infected file; hence, the infection confirmation function can determine the next course of action for the investigator (Blount et al., 2016).

Depending on the evidence collected in Phase 1, an investigator could take one of the following actions:

a) If the selected file is not infected, then the entire process can be terminated. The investigation can return to the initial process and the status can be declared normal;
b) If malware infection is declared, then a thorough analysis of the detected malware infection can be conducted, or the process of remediation can take place;

c) If the decision to analyse the infected file is made, then a record of investigation is necessary to obtain more information about the stages and the nature of the infection (Adeel & Tokarchuk, 2011); and

d) If malware is identified, then proliferation proceeds. It must be determined whether the detected malware matches any of the signatures or attributes stored in the database. If not, then this operation adds the new information to the database for future reference.

5.5.2 Phase 2: Detection and Analysis

Based on the results of the previous phase, files may be confirmed as infected by malware. Therefore, obtaining complete and accurate information about the phase activity and its goals is vital. Phase 2, as shown in Figure 37, identifies various attacks, clarifying whether the attack attempts to steal passwords, threatens the confidentiality of the data file, and so on. If it is difficult for an investigator to determine the rationale behind the attack and the consequences of the threat, then further analysis of the infected file is required to obtain details of the malware source.

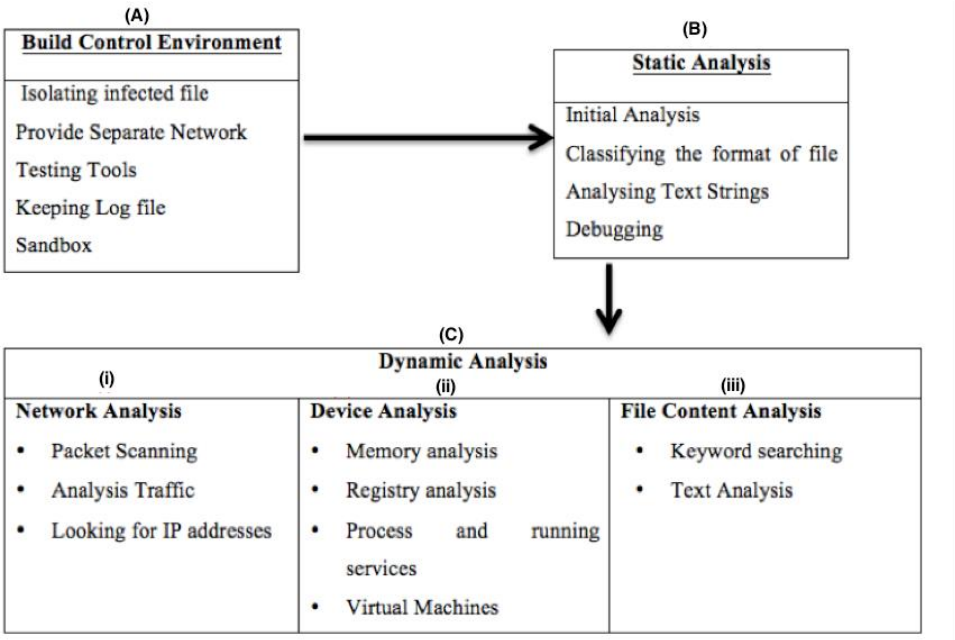


Figure 37 shows description of the second phase of TCMA (Source: author)

Detection and analysis illustrates the output functionality taken from Phase 1 (malware acquisition). After the investigator confirms the presence of malware, the captured malicious code is explored and its definitive characteristics are evaluated (Cavallaro et al., 2008). Defining and stating the attributes of the malware may aid the process of malware proliferation, which could further add information for updating and developing a fresh malware database (Phase 3). This is subcategorised further according to the variabilities described below.

(A) Building a Controlled Environment

Before further analysis and evaluation, it is important to develop a controlled environment for safer investigation. Safe investigation helps protect the environment from reinfection. This is a singular operative process, but with a dynamic environment and changes in requirements, the investigator must follow an ongoing analysis to achieve expected investigative goals and objectives (Chen et al., 2012). The fundamental idea behind the use of a controlled environment is to protect any system from the execution of a malicious code where it cannot harm other systems. This is typically achieved by emulating a real environment but restricting connections to it (SANS, 2009). The following proactive development is required for building a controlled environment:

- Secluding the infected file: Keeping the infected file away from live data is important. Infected files should be kept in quarantine to prevent any replication or reinfection in the network;
- Providing a separate network for investigation: During the investigation, it is important to select a separate network, including a separate ISP;
- Testing tools: All testing tools used for the investigative process should be clean before beginning the analysis. This aids in reducing complexities and confusion in later stages. This was extensively done by an auto-termination system, which restarts the tool every time a report is made;
- Keeping a log file: A log file should have special preference during investigation, as it carries the essentials regarding the activities performed on the infected file. Log files record each step for analysis, which can further ensure that the investigation is carried out according to specified rules and regulations; and
- Sandboxing for analysis: It is always preferable to conduct all analysis techniques in a

separate sandbox environment, as this makes it easier to perform evaluation without worrying about infecting memory files and other important system attributes.

(B) Variable Analysis

In this stage, the method and technique for malware detection that are to be used is decided. This method ensures the selection of one or more methods in this stage of malware detection. If an investigator chooses to go for multiple methods, then variable analysis provides a selection of different static and dynamic methods to carry out an investigation.

(C) Static Analysis

Static analysis is the most popular analysis technique and is performed on any suspected file system. All identified attributes can be used to add new information concerning malware proliferation and update the database for carrying out future detections. Malware always leaves traces and evidence; hence, a thorough investigation on executable files must be conducted through static analysis to predetermine future threats to the system. Each activity can be monitored and reports may be made in a variety of ways, including information on network accessibility and performance on the operation of the associated file system and internal storage. To optimise the static analysis process, the following guidelines should be followed (Cavallaro et al., 2008):

- Initial analysis: This involves investigation of the obvious symptoms of the attack to identify the nature of the infected file. First, the fingerprint of the file is matched to any existing malware profiles in the database (this is usually be done with MD5 hashing);
- Classifying the format of the file: In analysing files, each executable file contains schemata that may yield specific information about the attacker and the intention of the

attack. An analysis of information from menus, version information and calling functions could reveal a vast amount of evidence (Cavallaro et al., 2008);

- Analysing text strings: It is always important to scan the isolated text strings present in the infected file. Using various keywords in the built-in search field can help to retrieve the associated evidence; and
- Debugging: The last step for a static analysis is to conduct a process of disassembly. This helps to determine patterns, which can be helpful in identifying the attacker (Cavallaro et al., 2008).

(D) Dynamic Analysis

As previously mentioned, a static analysis is an initial activity performed by an investigator to gather generalised information on an infected file. As each piece of gathered information is important, the objective of the malware profiling must be maintained for the database to prevent future attack attempts. Moreover, a database can assist an investigator to work more efficiently in later stages. Although malware infection can be distinguished at premature stages, it is possible for unauthorised modifications to remain undetected. This can lead to compromises within the system after the attack. To obtain more details about the attacker, the dynamic analysis technique could be applied. Dynamic analysis typically gathers information on the following (Adeel & Tokarchuk, 2011):

- Memory: Malware often results in serious modulations of the buffer flow of the system and the critical programs and applications. Thus, a dense analysis of the volatile memory is required to obtain valuable information about the traces of malware activities and the intention to access the internal memories of the system;
- Exploring system registries: This process involves evaluating and performing an analysis on the changes occurring in the registry keys. The compromised system stores and illustrates

the exact modifications and activities that occurred on the system during the attack, which assist the investigator by easily providing evidence against the attacker;

- Investigation processes and running services: Investigating processes that were running in the system could show evidence of malware intention to start or stop particular processes intentionally. Analysing the running processes can illustrate the initial stage and the source's associated record of accomplishment; and
- Scanning for installed virtual machines: Investigating installed virtual machines in the system can assist the investigator in continuing the evaluation process. Malware could be inactive after virtual machines are detected, so the investigator will watch for the presence of virtual machines on the hardware to detect a virtual machine freely running inside the system.

Dynamic analysis is further differentiated into the following three types: (i) network analysis; (ii) device scanning; and (iii) file content analysis.

(i) Network analysis*:* A tremendous increase in the growth of malware varieties has made it increasingly difficult to detect each anomaly present in and around a system. There has also been a significant increase in the number of attack variables; therefore, it is important to deploy network security parameters to safeguard end users while coping with malware and traffic growth from both internal and external threats. Integrating a network analysis mechanism into a dynamic analysis is critical for malware detection, as networks are not vulnerable to malware practices that involve overthrowing and bypassing system security (Cobb, 2013); however, this will remain challenging for the investigator. A network analysis requires perfect synchronisation with other tools to be effective; if it is not properly tuned to the environment, this analysis can act too aggressively for mission-critical applications and other system processes.

Traditional malware detection methods only involve the database, which contains only signature marks or vendor-written scripts and protocols in which detection systems are isolated and

investigative pieces of malware are written and recorded. In contrast, embedding a network analysis for malware captures both known and unknown malicious activity by isolating all suspicious files and programs in a sandbox environment to confirm the detected behaviour (Keragala, 2016). In addition, the embedding network analysis tool can further help with investigating custom and polymorphic malicious programs often performed in APT-class attacks, as mentioned above. To improve the framework further (in later stages), a network analysis might be moved from a physical database server to a cloud-based server, which may help to reduce costs and increase the scalability and accuracy rates.

A network analysis was used to identify various types of malware-related attacks, such as analysing communication patterns between the compromised system and the attacker's control system. By analysing only network-based activities and parameters, such as protocols, ports, file extensions, type of content and time, an investigator should obtain a complete overview of the possible breach character, as well as information on the attacker (Celeda et al., 2010).

(ii) Device scanning: Device scanning is a simple but important tool in the proposed framework. The operating system is quickly scanned; hardware and installed programs on the computer are examined to discover any unauthorised activity. Device scanning will use process-scanning tools to check running rootkits, such as unwanted start-ups, executable files, maximum CPU-utilising programs, false APIs, and unwanted browsing and connections on the operative system.

(iii) File content scanning: Files are an important part of the system, as they carry data processed by the user and display any activity on the system before a compromise. Hence, file content analysis for the new framework is important, as malicious coding (also called Malcode) can be easily concealed, remain unseen in all extensible files and become embedded in executable applications. As such, an extensive approach is required to implement the analysis. Several tools are available that could be integrated into later stages of implementation to detect

hash files, identify file types and extensions, check for hidden Malcode, extract metadata for patterns, investigate anomalous content, extricate embedded files from the system, and so on (Fortinet, 2017).

One of the major issues concerning previous detection methods is the system's inability to identify ambiguous malware that is fully encrypted. In these scenarios, evaluating the content of the file is always recommended. File content analysis can be further justified by calculating the entropy of the file or evaluating the specific structure of the string in the file (Davis, 2009).

The method for calculating entropy can be employed to investigate files that are compressed, encrypted and modulated into various formats (Davis, 2009). For this scenario, the author has selected entropy calculation for any file system standards using the ENT - Pseudorandom Number Sequence Test tool, which evaluates results by comparing differences between the original and the infected files in a tabular format. This allows for static investigation and simple evaluation. In the string analysis method, it is necessary to search for the correct keyword. This is typically done through the International Centre for Automotive Technology (ICAT) tool from the Sleuth Kit (TSK) and the Auto Spy Browser, which works as a graphical user interface for TSK and employs search strings in the infected file (Dowling, 2016). The motivation behind using existing technologies was to integrate them in a more effective way into the framework to increase their utilisation and to get efficient results.

The key points for file content analysis are as follows:

- Analysing the aforementioned cases and samples could be tedious or complex for any malware detection system or investigator, so the proposed framework will integrate the file content analysis into a single interface, making for a simpler tool for analysts to use;

- File content analysis will help investigators to conduct analyses and store results in a structured database, which can be used to carry out further analyses (if required) of added file samples; and
- The focus of file content analysis is to detect file type and other extensions, e.g. Word tools for .doc and .docx files, portable document format (PDF) tools, etc. to respond in an effective manner.

This framework is an amalgamation of both static and dynamic methods, generating the best outcomes by mitigating all associated risks involved. As mentioned, the static analysis method assists investigators with analysing the distinctive features of executable files, while the dynamic analysis method focuses on the metadata retrieved from the infected file, including various parameters, such as the size of the file, its format and the information about the library management system. Both methods assist in detecting malware on the system. Compared with the standalone methods, the framework with the integrated approach provides a detailed analysis while mitigating all of the risks and vulnerabilities associated with the standalone static and dynamic methods.

5.5.3 Phase 3: The Database Operational Function

The database operational function focuses on database management, which is required to create an analysis report on the availability of malware in the system, depicted in Figure 38. The information carries various attributes about the detailed analysis of the malware. This information is important as it helps to update the existing profile or add a new attribute in malware proliferation. Any failure in the notification (Phase 1) could then be remediated to the operation of the database to monitor the occurrence of any reinfection of the malware in the associated system (Daryabar et al., 2011). Here, ‘remediation’ involves recommending a course of action in reference to the information saved in the database (if any) or logging the failure for further results.

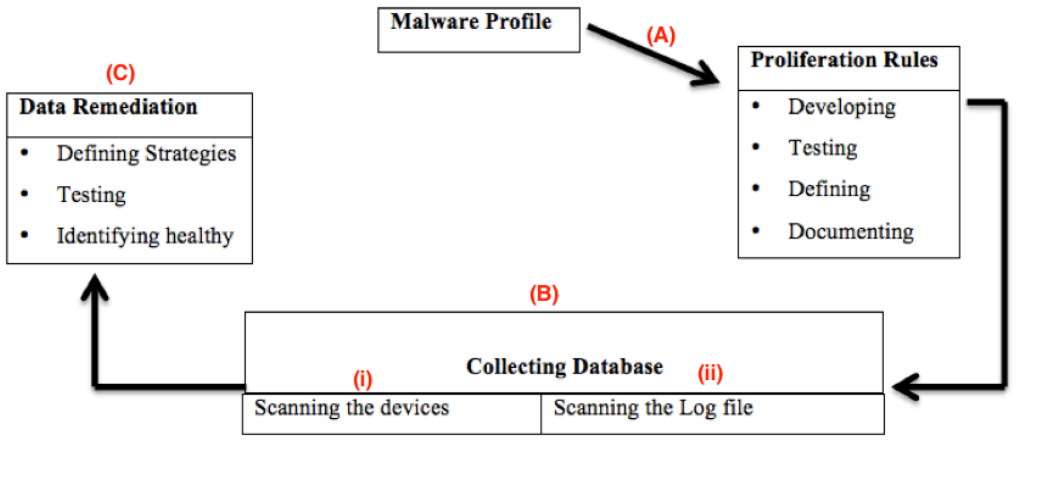


Figure 38: Description of the Third Phase of TCMA (Source: author)

(A) Building a Malware Profile and Defining Rules

Building a malware profile is important for future investigation. Samples from past records can be used to monitor, analyse and report infected files in the system (Trinius et al., 2009). A malware profile could assist in creating enhanced malware proliferation and preventing the possibility of reinfection in the future. A malware profile must follow rules to help achieve investigative goals. Following the dynamic and static analysis methods (Phase 2) allows the actions and manipulations of malware to be identified, which further assists in malware investigation. Malware profiling also includes the following:

- Summarising findings: Before conducting an analysis, it is important to summarise and categorise the findings in a single place. This can include compiling attributes from selected files, the system registry, running processes, running domains and open ports;
- Categorising: Classifying the profiles is convenient for other investigators to access information;
- Suiting the profile: Suiting the profile is accomplished through packaging processes in which analysts work out the rules of packaging and place all of the packaged material

into forensics tools to find inferences and associations for better results. This process makes the results more reliable for screening and scanning; and

- File sharing: All of the related findings should be stored in a way that may be shared externally in later stages with other business partners, including strategic vendors, conferences or mailing lists.

Repeating the process of detecting and profiling the malware periodically can prevent malware reinfection. Thus, an updated malware profile is another important function that must be considered to make Phase 3 more realistic and usable, as well as to ensure that investigative methods are robust.

(B) Collecting a Database

Collecting malware and managing the database is an important aspect of the database operational function. It involves the collection of new malware detected from the investigation in scanning the device and log files in the system. The database contains detailed information about the detected malware, such as its definition, the adverse effects of the malware on the system and the characteristics that will help to recognise the same malware in the future. During the investigation, if any of the files detected as suspicious match the stored attributes, the user can be alerted to the existence of the detected file. If new malware is detected through investigative tools, then the same database can be updated with new definitions. Detecting and scanning malware through the malware database could be an effective tool for investigations; however, this process requires regular updates on profiling so that users do not miss new malware in the system.

(i) Scanning Devices

Scanning the connected devices helps in testing the files for infection. Although scanning is a

traditional process for malware detection, new additions are being integrated to make it more robust. The new processes involved in the scanning process are as follows:

- A rule-based testing environment: In any investigation it is important to define scanning rules before initiating the testing process. Thus, it was important for the researcher to create a rule-based environment in the framework for a distinguishable initialisation of the process. The rules can be created based on the generated profile, and they can be saved in the tool itself to highlight various malware definitions;
- Analysing to the core: Once scanning is completed, it is important to analyse grey areas. It is possible that more than one system has been compromised, so it is important to carefully analyse and obtain the marker points, which can provide more clarity on the source device; and
- A rigorous documentation process: After the devices are scanned and a thorough analysis has been performed, documentation is created for future reference. Documentation can be as simple as writing a name with the IP or media access control (MAC) address of the device to create dissemination among the devices in the network. In sharing, such as using documentation for evidence in court, the documentation needs to be more formal and standardised. The most important thing to remember in documentation is that no assumptions should be made; everything should be based on data from the investigation. This process is completely dependent on the reliability and authenticity of a genuine and competent investigator following documentation norms, allowing for a data-driven approach based on facts rather than hypothesis.

(ii) Scanning the log files

Scanning the log files' content can also provide facts and clues that can be combined to make an investigation more effective and efficient. The following steps can be taken:

- Search for logged information: Similar to scanning the devices with tools, scanning log files can also work on the markers defined in malware profiling. Searching logs can be initiated by searching network logs and repositories, followed by scanning tasks, executable loaders, running processes and device logs in the system. Scanning log files provides an additional advantage in the investigation of inactive or unstable malware;
- Results analysis: To complete the investigative process, it is necessary to summarise testing through an analysis report. Creating statistics through data mining is an effective method to conduct an analysis on the system; and
- Documentation: As in the documentation process explained for device scanning, documentation can be carefully created to maintain usability, reliability and priorities.

(C) The Process of Remediation

The process of remediation is only effective in stages in which the investigation has already confirmed the presence of malware in the system. A decision then has to be made regarding the necessity of cleaning the malware. In some situations, remediation would be undesirable (such as in cybercrime evidence); however, once it has been decided to continue with the mediation process, wiping down the whole system, including the infected file, may be necessary. As modern malware is strong, starting the system over for remediation surety is always recommended. The effectiveness of remediation cannot ensure the cleaning of the infected file or malware, so it is always better to test remediation effects immediately after the process is complete.

It is very important for every investigation to have an effective and efficient model for detecting and analysing malware. As all investigations need a comprehensive approach to finalise the result, it is important to have an integrated model where all functionalities should be linked to one another.

In Tier 1, the malware acquisition function, the main purpose is to confirm whether or not the given malware is infected. If the file is discovered to contain a malware infection, then the whole process will be shifted to Tier 2, malware detection and analysis.

Tier 2 will use the hybrid methodology (employing both static and dynamic methods) to perform comprehensive analysis. Each result will then be shifted to Tier 3, database operational function, for malware proliferation and data remediation.

5.6 Summary

The current chapter proposed and described the rationale behind an effective malware detection and analysis framework. This framework was developed based on market research carried out in Chapter 4 where various rational flaws that exist in the investigative processes for detecting malware in information systems were highlighted by current users. Moreover, upon further investigation, vulnerabilities and limitations were found in existing forensics tools for investigating malware, which were illustrated based on specific parameters, such as recovery, usability, performance, functional operations and the reinfection detection process. This chapter progressed the identified limitations in the investigative processes where a critical comparison was performed between signature-based, specification-based and behaviour-based detection methods, focusing on the limitations in their accuracy, performance and functionality. Thus, a need was identified to develop and deploy a new system that would assist investigators in the accurate and complete detection and analysis of malware without compromising data integrity. A new framework called the TCMA (tri-CORE malware analysis) method was subsequently proposed. The three (tri) tiers within this method are based on three distinctive phases of detection and analysis in which the entire research pattern is divided into three different domains. The tiers can be defined as the malware acquisition function, detection and analysis, and the database operational function. The model could provide a new advantage in the field of

digital forensics by reducing the cost of buying different tools, providing accurate information and increasing the throughput compared to other existing investigative tools. New functionalities were integrated and measured (by critically analysing related literature and corresponding market analysis results) before being put into the framework, making it more advanced, efficient and effective than any existing models for malware detection. New parameters such as recovery, usability, performance, functional operations and a reinfection detection mechanism were introduced, which were seen as restrictions in the current frameworks (ref: Table 7). Consideration was also given to the comparisons between current malware detection methods on the basis of accuracy, performance and operation rate to conduct a critical review of their performance (ref: Table 8).

New functionalities, such as introducing network analysis as a part of dynamic scanning, give the new framework an advantage. Network analysis for malware captures both known and unknown malicious activity by isolating all suspicious files and programs in a sandbox environment to confirm the detected behaviour; in contrast, traditional malware detection methods are typically only based on the database, which contains signature marks or vendor-written scripts. Even in terms of scanning, the new rule-based method and documentation process adds more ease and accuracy in malware investigations. Nevertheless, the proposed framework has contributed to the field of IT security and forensics and has also helped in collecting the artefact (Toolkit) requirement, development and testing (discussed in Chapter 6) for the existing study. The critical functionalities from the proposed framework will be utilised to define a simple yet robust integrated open source malware detection software, which will help the investigator develop skills to critically respond to the cyber incidents with flexibility in any required environment.

CHAPTER 6 - ARTEFACT DEVELOPMENT

6.1 Overview: Building a Malware Analysis Toolkit

Exploiting the capabilities of malware detection software allows investigative teams to explain and declare the nature of the incident, which can reduce further infections in the future. In this chapter, it is the author's intention to develop an artefact that reflects the critical functionalities of the proposed framework, as outlined in Chapter 5 – a malware analysis Toolkit, which is simple in structure yet sufficiently robust using integrated open source malware detection software tools. This Toolkit is a prototype, which can be built into commercial software later for the purpose of helping investigators develop skills to critically respond to current cyber incidents flexibly and in any dynamic forensic environment.

It will support the critical needs and requirements by:

a) Supporting an investigator in analysing given malware;
b) Ensuring security during the investigation through robust authentication and authorisation methods;
c) Performing a rapid request and response of results;
d) Creating effective reports on the findings for further analysis; and
e) Delivering a usable and professional platform for law enforcement.

The development follows prioritisation, reflected from the proposed framework – TCMA (Tri-Core Malware Analysis), as described in the previous chapter (Chapter 5). The ***critical values*** that can be deployed in the Toolkit have been identified as:

a) Minimising the number of investigative cycles;
b) Maintaining a safe and controlled environment for investigation;

c) Performing both static and dynamic investigations through proactive open source tools;
d) Ensuring incident response situational analyses are efficient;
e) Using a single platform for multiple malware analyses;
f) Implementing/deploying only effective tools, reducing the usage of irrelevant tools;
g) Save investigated data for future usage by accumulating databases;
h) Minimising the cost of using different tools by amalgamating into a single tool for investigation;
i) Generating a cost-effective reporting structure; and
j) Decreasing the frequency of paying additional investigators for the same work.

6.2 About the Toolkit

The Toolkit is an effective malware detection system comprising multiple accesses to various open source malware analysis tools. Open source code, whereby the source code is made available for any person to change and distribute the software for any purpose (Andrew, 2004), is common in malware security-based software, because it allows more people access to help develop it, aiding the development in a rapidly changing environment (Simmons, 2016). The selection of open source software for this Toolkit was a conscious effort to minimise upfront costs with this development cycle, and was also based on the survey results from Chapter 4, the market report and analysis, where respondents gave a preference for the use of open source code. Recommendations and preferences were collected through two research methods, namely the *conceptual work* and *current analysis.* The conceptual work focussed on data collected by analysing literature related to the study of malware and its detection processes by various scholars and researchers, while current analysis was performed using a primary research method (PRM). The PRM included the questionnaires and interviews (refer to Appendix A), involving experts and practitioners related to computer security, cybercrime investigation and malware detection methodologies. A thorough analysis was performed to

establish the outcomes. It was suggested in the results that tools should be used that employ both static and dynamic methods for detection (Chapter 4 and Appendix B). The Toolkit is capable of carrying out processes that can assist in analysing the infected files and checking the optimisation of open source tools available worldwide. A rationale was designed based on certain parameters, illustrating the reasons for using the Toolkit over a single tool for malware investigation. These are presented in Table 10.

Parameters	**Generalised Single Tool**	**Designed Toolkit**
Flexibility	If a malware tool does not allow any form of modification in its design or method, this inflexibility could detract from the user's quality experience. Such tools are proprietary software and they do not offer any form of modification as they are restricted and, therefore, suffer from this trait.	The Toolkit takes a flexible approach to customisation and allows changes to the design or its methodology to suit individual user requirements.
Scope	A single tool that has its own particular application and is not part of a suite of tools; in essence, it has a restricted scope and, therefore, has limited usage.	The Toolkit has a feature to add new tools or delete existing tools at any stage as required by the user. This offers greater scope to apply to a range of applications, not just a single one.

Security	Most tools come without an authentication and authorisation process. Any user or investigator can use the tool anonymously, thereby increasing the risk to integrity. Forensic tools such as Wireshark or Registry Monitor are desktop-based and can only be installed on the same platform, with reports being made only on that platform.	The Toolkit comes with a two-user authentication system (Super Admin and User). All activities can be monitored through the Super Admin account, thereby producing an added level of security.
Reporting structure	Some tools can only provide results to be analysed by an investigator. When an investigator is using multiple tools, he or she needs to follow a multiple tool format based on that tool to make reports. With every new tool, a new format is needed for report creation. No specific or standardised reporting formats are available, making knowledge transfer difficult and requiring prior knowledge for use.	The Toolkit possesses its own reporting structure functionality in the form of screenshots. This makes a reliable, but simple output for report creation. A single style of reporting can be shared and read in same format.
User interface	Each specialised tool comes with its own user interface. This increases the chances of a less user-friendly environment occurring with multiple	The Toolkit has been designed with the focus on developing a user-friendly environment. The environment is also fully

	interfaces, and requires the knowledge and experience of a multitude of different environments.	customisable which allows users to change settings to their preference to aid in setting a scene that an individual user will be familiar and happy with.
Usage	A single specialised tool is intended to be used on a specific scenario and environment. They are not intended for usage across a wide range of applications as they have dedicated roles.	The Toolkit comprises an amalgamation of various tools present in various categories. It can be used in any dynamic environment; this offers more usability as its application to different possible tasks is increased.

Table 10: Comparing Generalised Single Tools with the Designed Toolkit

Further highlights of the developed Toolkit are as follows:

a) It creates reports on the findings for further analysis;
b) It has an enhanced GUI for investigating multiple entities;
c) It decreases the cost of using different tools for investigation;
d) It reduces the number of investigative cycles;
e) It uses a single platform for multiple malware analyses;
f) It is efficient for incident response situational analysis;
g) It decreases the need to pay additional investigators for the same work;

h) It implements only effective tools, reducing the usage of irrelevant tools. Here, optimisation refers to using only those tools that are required in a given environment while minimising the inappropriate ones;
i) It saves investigated data for future use in databases; and
j) It provides a cost-effective reporting structure. When a similar situation occurs, an investigator can seek results from the database and can update the previous records, preventing any sort of redundancy

Before initialising the development, it is vital to seek answers to some important aspects of the work that reflect user notions and conceptions in the field of forensics investigation, such as:

a) What actions can an investigator take with respect to an infected/ineffective system?
b) How can evidence against attackers be obtained?
c) How can a detection process be differentiated based on the nature of the attack?
d) Is a dynamic solution needed for every scenario, or can all investigations be performed with a single tool?
e) How is the number of tools required for an investigation determined?

These questions can be addressed only if an investigation is undertaken in a controlled environment and by performing a rigorous analysis (Lindorfer et al., 2011). To justify the probability of obtaining answers to the abovementioned questions, a prototype has been proposed in the form of a malware detection and analysis Toolkit. The objective of developing the Toolkit was to reflect the new TCMA framework and propose a live implementation of a malware detection system by integrating open source tools for testing. For developing the artefact, studies and observations from previous chapters (Chapter 2, Chapter 4 and Chapter 5) were considered. It was vital to emphasise a dynamic environment to match users' expectations and requirements; thus, a market research document was included as the key activity. This

results in a more definitive problem statement by providing a specification document to model software development activities. Once the artefact was developed, this executable system was tested under a controlled environment for observation and analysis (in Chapter 7). Figure 9 shows the relationship between previous research activity and Toolkit development.

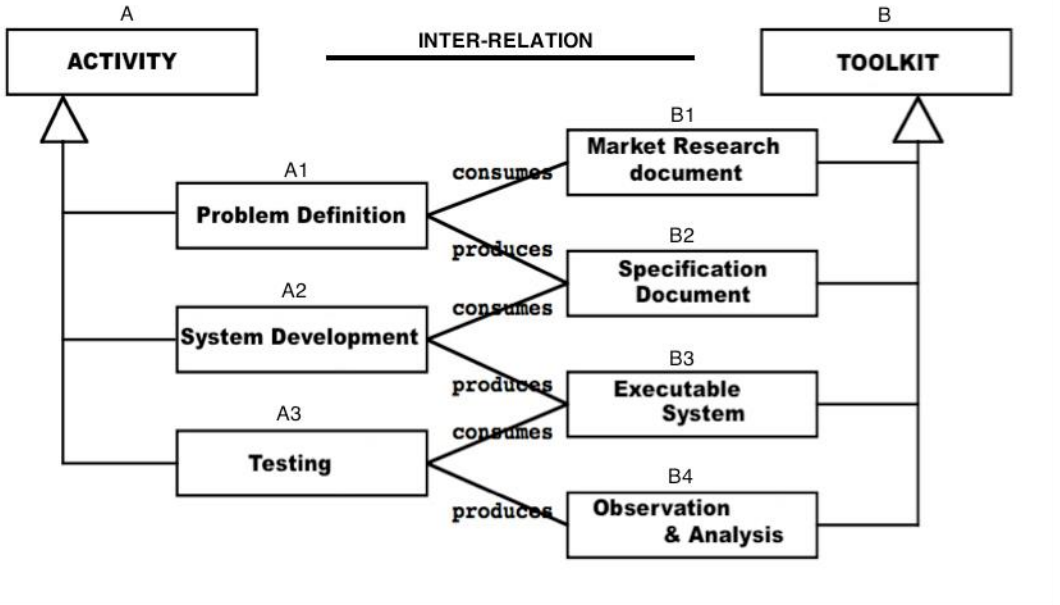


Figure 39: Relationship between Research Activity and Toolkit Development (Final Artefact) (Source: author)

6.3 Artefact Development Model

There are various software development lifecycle methodology models available for developing a software-based artefact and, in practice, selected based on the aims or goals of the project outcome. This methodology will set out the guidelines for the approach upon which the software will be designed, built and maintained. Although various different models do exist, the most commonly used are the Waterfall model, the Spiral model and the Iterative model (Alshamrani & Bahattab, 2015). Each methodology will have its own particular strengths and weaknesses, but all follow a format of sequential steps to allow for a system to be developed.

The Waterfall model is a widely known and, therefore, used model where it is considered intuitive although less practical. However, all other lifecycle models are derived from the Waterfall model (Ruparelia, 2010). As the most widely known model, it is used extensively throughout industry (Petersen et al., 2009) and incorporates a logically applied series of steps. These steps lay out the process in which development occurs; each step is completed prior to starting the next one, although this leads to a less flexible methodology.

The Spiral model has its place where flexibility is demanded. The project is passed through the same phase, or spirals, until it is completed. This allows for multiple refinement opportunities, incorporating feedback and, subsequently, making further improvements. The Spiral model is highly personalised due to this feedback loop being directly affected by the end user (Barjatya et al., 2017). Alternatively, when a flexible approach is deemed necessary, an Iterative model can be incorporated where guidelines are used to help pinpoint the software requirements by testing and evaluating through different iterations until each phase is satisfactorily completed. This allows a version to be completed relatively quickly, and depending on the required iterative loops, a solution can be rapidly found, although conversely, too many loops can result in a large resource drain (Ghahrai, 2008).

In practice, any methodology can be applied to any development process but will carry a different level of risk, budget, timeframe, and particular benefit to any specific project. It is ultimately up to the developer to examine the strengths and weaknesses of any particular methodology and assess these in relation to the project that is to be undertaken.

The main constraints when the author selected the software development model were based on the general domain in which it was to be used, and the timeframe that was deemed suitable for

learning and using the model. Based on these constraints, the Waterfall model was selected as the standard model for development.

The Waterfall model was selected as this is one of the simplest and easiest to understand; and given the time constraints of this project, these were important factors in adopting a model while ensuring an adequate design methodology could be put into place. The Waterfall model is also better suited to smaller projects where requirements are well understood and remain less dynamic during development, which was well suited to this research domain. Figure 10 shows the systematic diagram for a Waterfall model for developing a malware detection and analysis Toolkit.

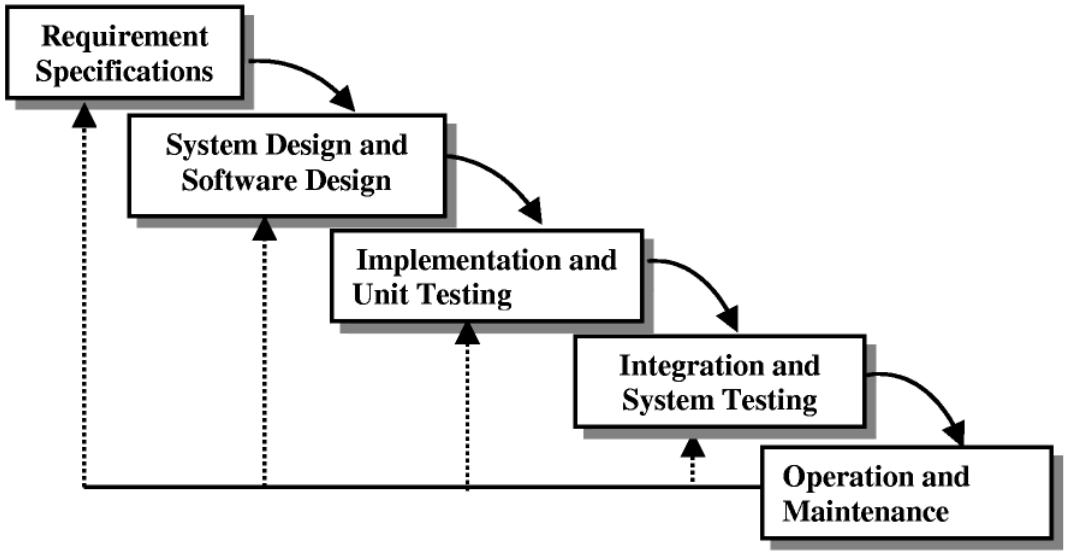


Figure 40 Systematic Diagram of a Waterfall Model

(Waterfall Model , 2009)

Before initialising the process of gathering requirements (Phase 1) and other needs for the artefact development, it is important to highlight the problem statement and acknowledge the challenges associated with a malware analysis. The malware spectrum is extensive and it is important to rigorously analyse it to fully understand the complete threat. No single platform is

100% threat-proof or immune to attacks (Mell et al., 2005). As discussed previously (Chapter 2), modern malware is multi-layered in nature and possesses many infectious vectors and other payloads to make an infection harmful. Traditional detection methods are often not adequate in malware detection due to framework customisations. Moreover, attackers are using anti-forensic techniques to obfuscate infection and prevent detection (Garfinkel, 2007). Thus, considerable time, expertise and knowledge domains are required to quantify modern forensics techniques to enable robust detection.

The two main techniques used by investigators to detecting malware are static and dynamic analysis. Static analysis focuses on investigating the code without executing it. Investigation is undertaken using dissemblers, searching the string, API callings, library calls and graphs. It reconstructs data structures and forms a union with the source code to draw conclusions (Sponchioni, 2010; Tratt, 2011). Although the static technique provides valuable information for investigation, it is time-consuming and does not allow anti-forensic techniques in the form of packers, which are commonly used by attackers, to be detected (Sponchioni, 2010). In contrast, dynamic analysis does not focus on source code at all; rather, it highlights the observations performed on hosts and networks. It uses mechanisms to investigate the use of registries, files and other monitoring tools (Egele et al., 2010). Investigators prefer this technique due to its flexibility and usability. Still, using these tools, malware can change its behaviour after being detected. Malware can delete itself, decide not to run on a payload or decide to destroy/harm the system (Vemparala, 2015). As mentioned in several reports (Damodaran, 2015; Tzermias, 2011; Zeltser, 2010, 2015), it is more productive to use iterative and recurring techniques, which use both static and dynamic tools for an investigation in significant detail (Damodaran, 2015; Moser et al., 2007; Tzermias et al., 2011). In the proposed prototype, static and dynamic techniques will be used in conjunction. Concatenating static tools will be used to detect malware proliferation, and dynamic tools will be used to focus on behavioural detection and signature

analysis, which have been shown to provide an advantage, yielding an effective and efficient investigation.

The selected analysis process for artefacts involved a one-dimensional simplistic approach for performing malware analysis, as shown in Figure 11. This method was selected for its simplicity and ease of implementation to allow for a quick but robust approach. Figure 11 illustrates that the malware sample is one of the two inputs to the methodology with a report created as an output. The result produced can create feedback, which would then be converted into a report format again for improvements through a post-analysis assessment.

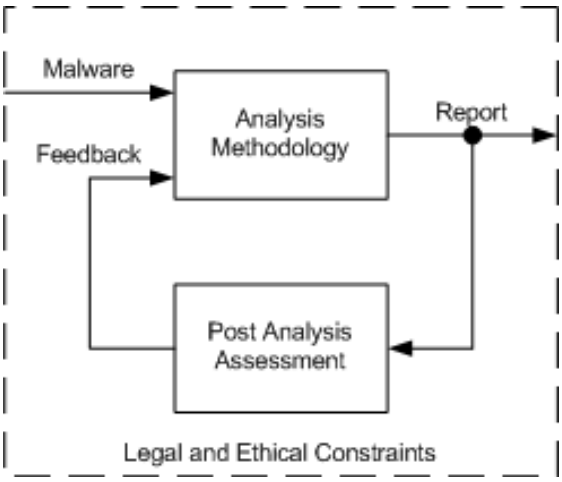


Figure 41 Selected Analysis Approach for Performing Malware Analysis

[Source: author]

6.3.1 The Requirement Gathering and Analysis Phase for Toolkit Development

The previous chapters illustrated the findings and associated issues in the study of malware development and analysis. It was highlighted how vulnerabilities and limitations exist in the current forensic tools for investigating malware, surrounding the recovery, usability, performance, functional operations, and reinfection detection processes. Using these vulnerabilities and limitations, the value of developing a new system that would help

investigators to accurately and completely detect and analyse malware was identified. This new system would need to reduce the costs of multiple tool purchases, provide accurate information and increase throughput over existing systems. New functionalities to increase the overall efficiency and effectiveness were also recommended and would be of further value to a finalised product.

Considering the critical analysis from the literature review and observations from the market analysis, needs and requirements were gathered. Development propositions, including system planning and details on the execution, which were projected from the proposed framework (TCMA as mentioned in Chapter 5) are listed below.

User Requirements

a) Generic Functions

- The system shall allow users to perform *multiple investigations* through the authentication access;
- There shall be two user roles: Administrators (Super Admin) and Investigators (Users);
- The system shall allow users to *generate reports* on each activity performed;
- The system shall allow Super Admin to add/delete *Investigators (users)*;
- The system shall allow users to *save results* in the database, which have been detected live from the investigation in the user-specified folder; and
- Administrators shall be capable of adding/deleting/modifying/viewing any user activity.

b) Security

- The system shall apply secure login IDs and passwords to provide a robust authentication and authorisation mechanism for secure investigation. The self-designed

system uses security protocols and encryption. Limited access shall be given based on the type of user; and

- The system shall ensure protection through high-end backup in a given database for each record and result generated.

c) Portability

- The system shall use a programming language which is platform independent. It shall also provide an efficient way of developing and running desktop-based applications on any operating system, making it portable for use on any configuration.

System Requirements

The system will have two categories of users:

- Super Admin: Super admin will have all rights to add/delete/modify any system configuration or functionality; and
- Investigator: Investigator rights will be limited, depending on the case, scenario and designation. Investigators can create reports and databases. To ensure data integrity is maintained and only the investigator can access their own specific cases, the database will be separate for all individual investigators.

The main functions of the system, which includes user profiles based on the access control mechanism, are as follows:

- Login;
- User creation;
- The adding/deleting tool category;
- View list of users;

- View list of tools;
- Upload infected file/linking network;
- Report generation;
- Results; and
- Logs/last records.

Technical Requirements

a) The software requirements are as follows:

- Win 7;
- Oracle VirtualBox;
- MySQL; and
- .NET (dot net) Programming

b) Considering the required software, the malware samples to be used in the system, and using computing resources capable of running similar applications, the following hardware is required:

- PC/laptop (any platform);
- Hard disk (minimum 50 GB); and
- RAM (4 GB).

6.3.2 Design Phase for Toolkit Development

The following section shows data flow diagrams and describes the general methodology, displaying a graphical representation of how data is passed through the Toolkit. Use case diagrams are also provided in this section to describe the actions of the Toolkit system. Also

included is an activity diagram, which describes the dynamic aspects of the overall system to provide a graphical display of activity-to-activity connections of the Toolkit.

Data Flow Diagrams

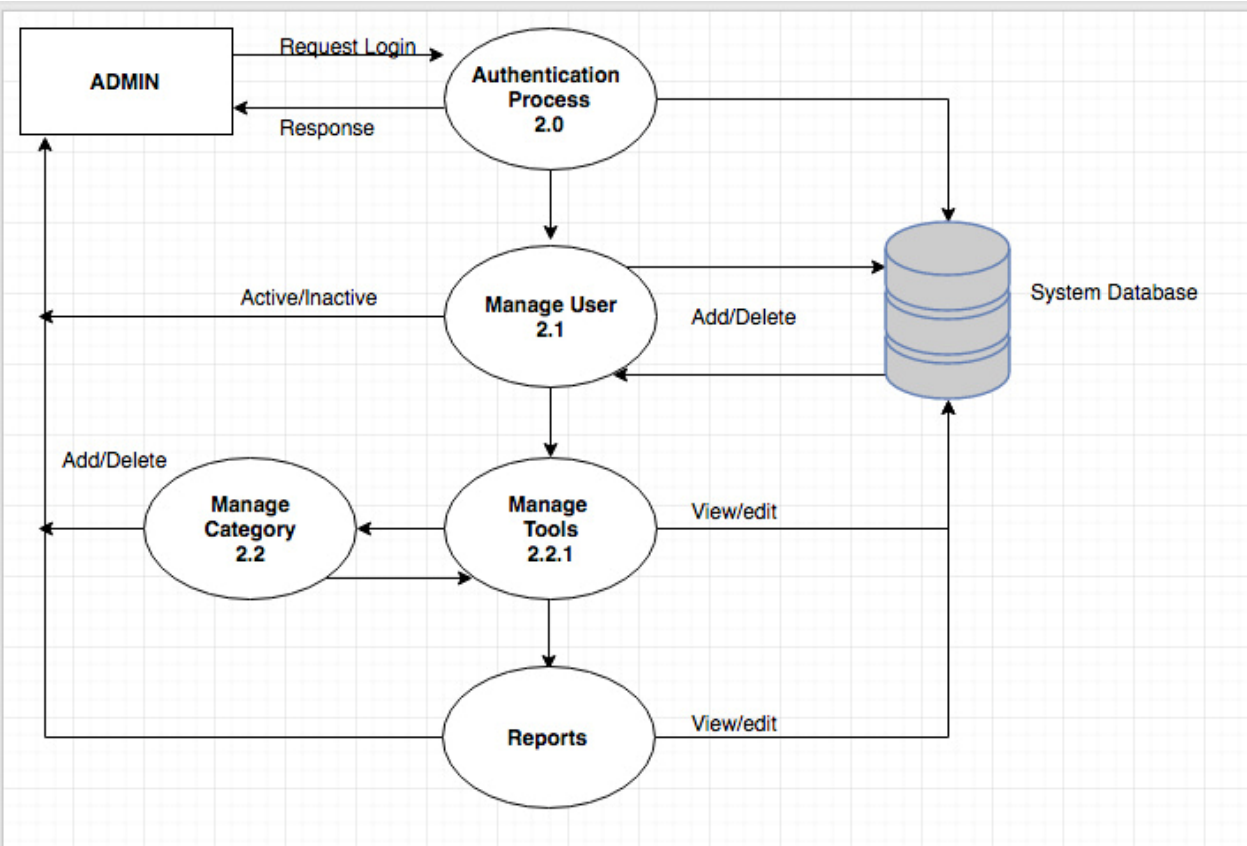


Figure 42: Data Flow Diagram for Admin Panel

[Source: author]

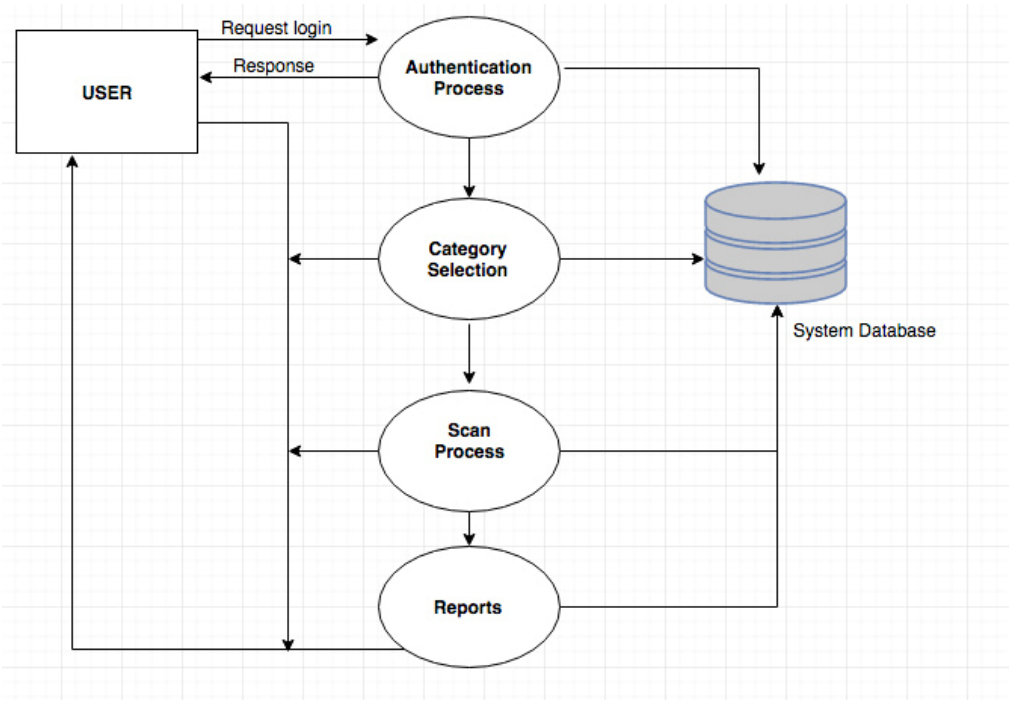


Figure 43: Data Flow Diagram for User Panel

[Source: author]

Use Case Diagrams

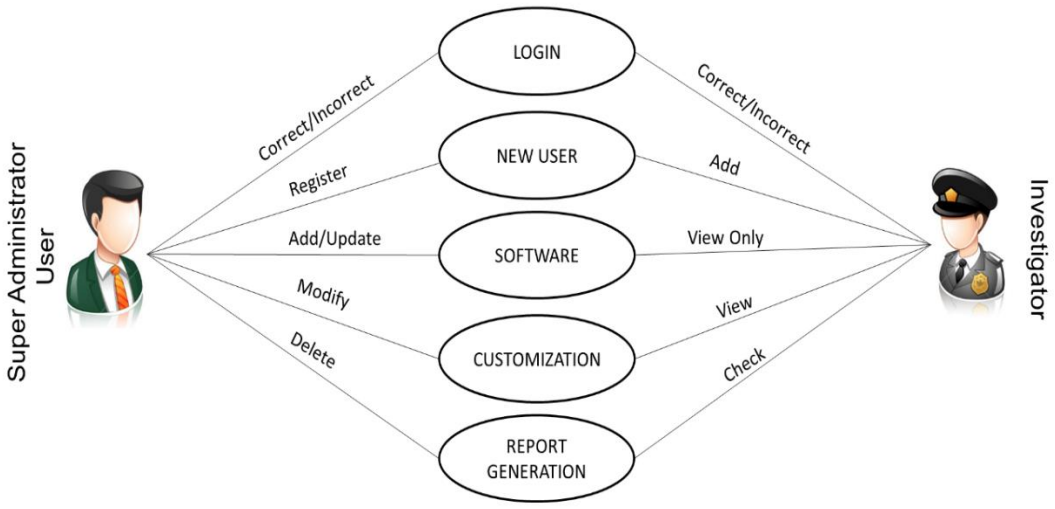


Figure 44: Use Case Diagram Illustrating Working System

[Source: author]

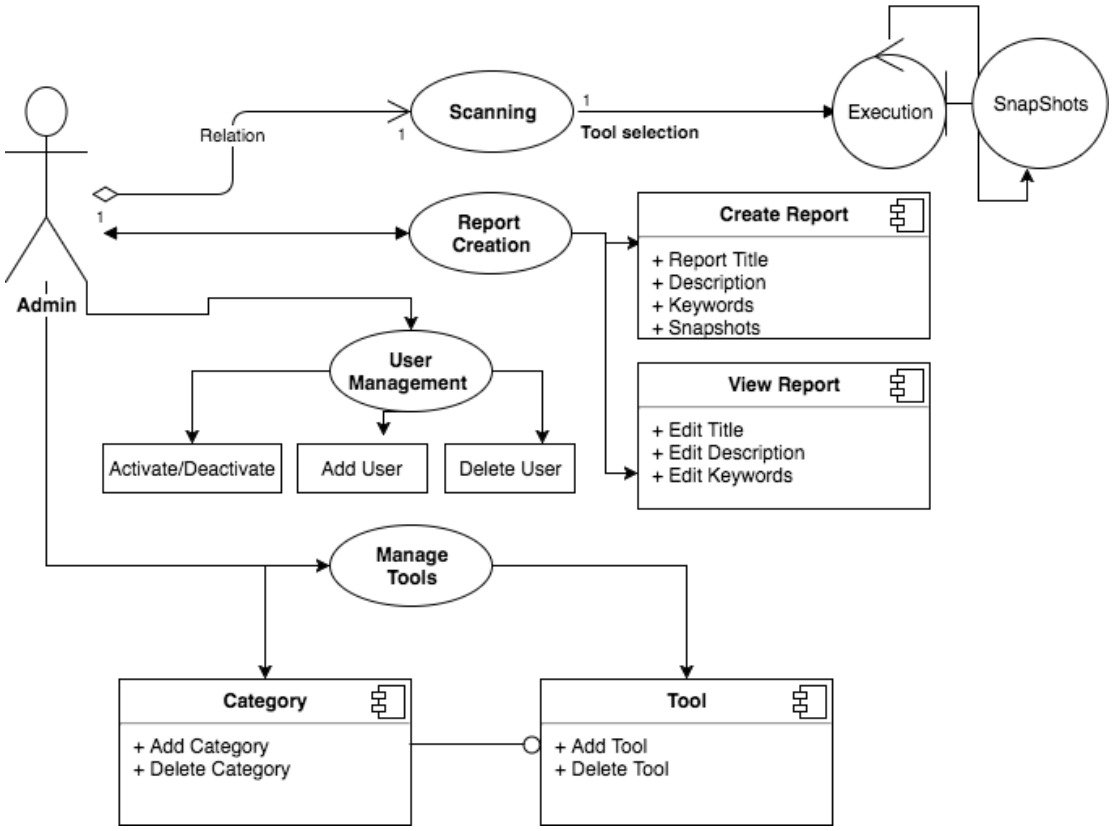


Figure 45: Use Case Diagram for Admin Panel

[Source: author]

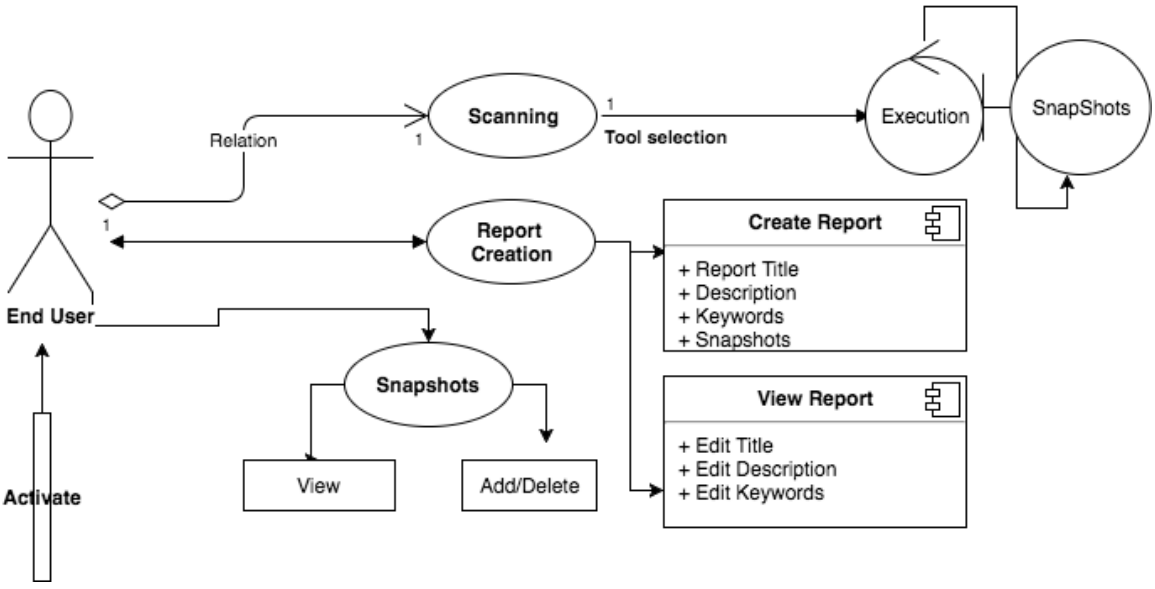


Figure 46 Use Case Diagrams for User Panel [Source: author]

Activity Diagram

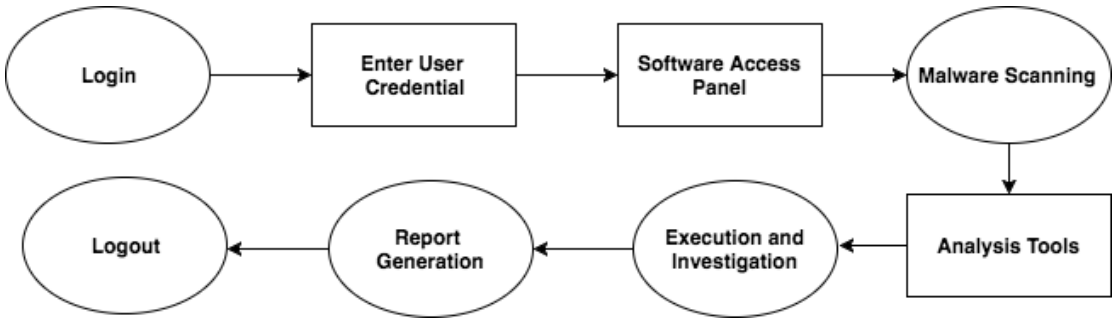


Figure 47: Generic Activity Diagram for System Implementation

[Source: author]

6.3.3 Implementation Phase for Toolkit Development

For implementation, essential hardware and software were acquired and implemented. This section explains the implementation of the Toolkit, which can help in developing a malware detection toolkit in a more robust and efficient way. It explains various stages for the development of an integrated platform that functions as a toolkit for performing an efficient malware analysis.

The entire process of implementation is divided into two investigative domains:

A. The pre-implementation proposition; and

B. The post-implementation proposition.

A. The Pre-Implementation Proposition focuses on the activities that include the building of an integrated toolkit acting as a framework (working model) for analysing malware activity. This can be divided into two phases, as follows:

- Phase 1: Working on the back end; and
- Phase 2: Working on the front end.

As the back end possesses the main functionality of the system, this is the phase that is developed first. The back end is the brain of the artefact. This is located on the backside, and is not visible to the front-end user. It is built with a server side language and database. A back-end application will return results to the user via front-end queries in the form of buttons [126]. The front end is usually an interface for user communication with the system in a format supported by a GUI. This part of the development is visible to the user via a suitable style and material. Here, development was made according to the designs and the integrated application source code, and then making the code adjustable for back-end services. In addition to the source code, the researcher used several predefined libraries and frameworks to make the development more simplistic and less time-consuming (Fender & Young, 2013).

In developing the artefact, the .NET framework was selected for front-end development while MySQL was selected for back-end implementation. The rationale for selecting the preferred languages used for developing the front and back ends of the artefact is as follows:

Using .NET as a Platform

The author selected .NET in the Toolkit implementation because it is encoded with high-end security codes that provide secure executions when compared to the traditional languages, such as C and C++, which are rigid and restricted in their functionality. .NET provides faster and safer executions compared with the other source code languages. This programming language is one of the most trusted languages across the globe for developing secure, robust applications

that are easily extendable to market needs. This ultimately ensures investment protection at a lower cost (Taboada et al., 2009).

Various attributes make .NET one of the best languages for developers to create desktop-based applications, especially those that require dynamism in the operational environment. Some of the critical attributes for selection are as follows (Baitsch, 2010; Dickens & Thakur, 2000):

a) **Rapid Execution:** .NET provides a rapid platform for executing the source programs faster, as the codes are effective during program usage. No other traditional programming language provides executions that are as rapid (Microsoft.net, 2017);

b) **Robust Programming Language:** .NET is a language that provides a robust programming platform, which is aimed to deliver secure, high-level programming code implementation. .NET is among the latest and smartest tools for development; it delivers the most trusted applications compared with other languages, which offer fewer and less secure applications compared to the .NET platform;

c) **Massive Machines:** The .NET language is commonly used in large systems that require high security and complex functionalities. .NET is a secure language that is easy to maintain at a low cost. The language is often implemented in large organisations to secure back-end operations (Sheriff, 2017);

d) **Flexible Operations:** .NET is flexible in its operations and interface, as performance enhancements can be easily accepted without changing the structure of the source codes. .NET virtual machines can be deployed in current applications without recompiling the existing codes. Technology hardware configurations and operating systems can be advanced without making any complex changes in architectural interfaces;

e) **Performance:** Programming languages can be compared based on performance, which can be evaluated at the time of using any developed applications. The .NET language creates a benchmark for delivering the speediest performance amongst other languages (Obasanjo, 2013);

f) **Ease of Development:** An important factor for any developed application is how user-friendly the application is, and any good application must be embedded with a good quality code and a debugging functionality, which is embedded in all .NET applications (Obasanjo, 2013);

g) **Ease of Maintenance:** Maintenance of any developed product must not be complex, as it will be difficult to use with complex functionalities; the .NET language is best for providing the core functionalities, such as monitoring and error diagnostics; and

h) **Portability:** The .NET language is built using the Common Language Infrastructure (CLI), a language-neutral platform that allows functions within .NET to not be specifically tied to a single developmental language. This allows for modifications, upgrades and source code adaptations written in various languages to be used in .NET, allowing for a wider audience of code developers to easily implement change.

Using MySQL as a Database Tier

MySQL is a fast, reliable database system based on relational methods. There are many database platforms, such as DB2 and MaxDb, and amongst them, MySQL is considered the most trusted platform for managing database operations (Lee, 2013).

The basic characteristics of MySQL justify its selection for a database tier. Some of the essential attributes are as follows (Kulshrestha & Suchdeva, 2014):

a) **Supports the Client/Server System:** MySQL is a standard database procedure that is based on the client/server architecture. The client/server architecture is a standard

security system; it is responsible for maintaining the security of the data, as client and server data can be maintained on different servers, and the information is shared on a secure platform. Other database companies, such as Oracle and Microsoft, also support the client/server architecture, but they focus more on file servers. The drawback of using the file server system is that it becomes overloaded as the number of clients in its database tables increases;

b) **SQL Compatibility:** SQL stands for 'standard query language'; this is responsible for running statements such as the data manipulation and the data control language. These are the standard statements for retrieving and updating the contents of the database, and MySQL supports these operations;

c) **Platform Independence:** A MySQL database can run on a variety of operating system platforms, such as Windows, Linux and Macintosh, and it supports UNIX variants as well. Thus, MySQL is considered platform-independent, making it an international platform for managing the back-end operations of worldwide applications; and

d) **Fast Executions:** The MySQL database supports fast executions of updating and retrieving the content in the database, as the source codes are efficiently embedded in the database coding. The MySQL coding tests a variety of platforms, thereby rapidly retrieving database operations.

B. **Post-Implementation Proposition** focuses on the activities related to individual testing of the tools in a controlled environment. Selecting an appropriate tool is vital, as it assists in making a powerful yet cost-effective instrument for analysis. A scheme was devised for different phases of developing a controlled analysis lab and was based on the work outlined by Zeltser (2015). The development was conducted through the phases listed in Table 11.

Phase	Task	Example
1	Allocating a physical or virtual system for the analysis laboratory	Oracle VirtualBox
2	Selecting a behavioural/signature network/process analysis tool	Process Hacker, Wireshark, etc.
3	Testing the tools on sample malware	Running the system in a safe environment and then running the selected malware to capture results
4	Analysing the recorded data	The result is recorded and saved in the database selected by the user
5	Results and conclusions	Analysis, evaluation and reports can be formulated while investigating the captured output

Table 11: Phases of the Post-Implementation Proposition of the Toolkit

6.3.4 Testing Phase for Toolkit Development

Testing is an important phase in the development of software as it executes and runs the system to observe it working. Before initialising this phase, it is important to create suitable environmental conditions for running the system. For this, an infected system was required which can be further investigated to generate high-end results. Furthermore, it is essential to select appropriate tools for conducting successful testing investigative processes for clearer

observation later (in the next chapter of this thesis). This section is divided into two phases: Phase 1 discusses the attributes for developing a safe analysis laboratory as an environment for conducting investigations; Phase 2 discusses the selection of open source tools for integrating the Toolkit.

Phase 1: Allocate physical or virtual systems for the analysis laboratory.

A common approach to examining malicious software involves infecting a system with a malware specimen, and then using appropriate monitoring tools to observe how it behaves. This requires a laboratory system that can be infected without affecting the production environment.

The most popular, flexible approach to setting up a lab system involves virtualisation software, which allows the use of a single physical computer for hosting multiple virtual systems, each potentially running a different operating system.

For the present development, the author used Oracle VirtualBox. VirtualBox is an open source proprietary licence, making the research more cost-effective than many other virtual software; moreover, for academic purposes, especially with a Windows desktop, this virtual software is robust and reliable. The following are some important points that should be considered when allocating virtual systems for a safe environment:

a) Deploying and running virtual environments together on a (isolated) single machine is essential and vital for analysing malware in which interactions and analyses from other systems, obfuscating data and other instructions from hackers or updating systems are required. In all situations, virtualisation can be easier and more flexible for users than installing numerous sandboxes (Broder, 2012);
b) Another very useful capability of a virtualised environment is that it can take instant snapshots of the complete system, thereby working as a laboratory for investigation.

This can help to possibly record the status of the system before and after the infection, and then to remediate the environment back to normal with a single click of a button for analysis (Distler, 2007); and

c) During installation and while using a virtualisation environment, maximum random access memory (RAM) is installed and kept in the physical system, as this can represent the most important factor in increasing the performance of the analysis. Moreover, a larger hard drive is installed to host several virtual machines on a single system, allowing it to store the maximum number of files for investigation (Broder, 2012).

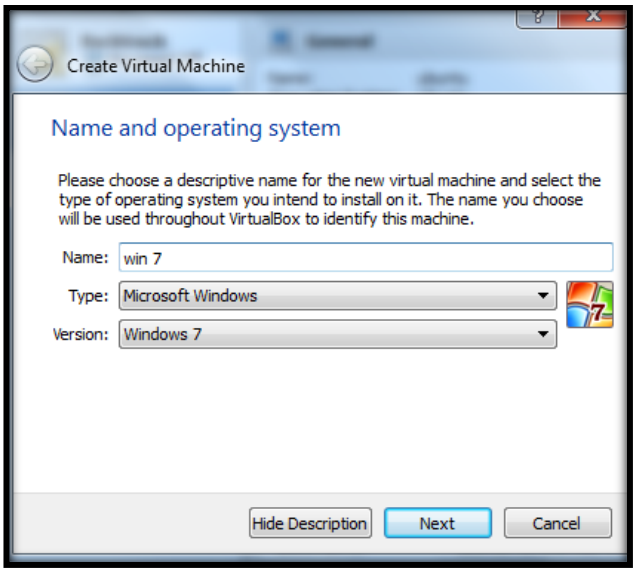


Figure 48: Screenshot of VirtualBox Installation (Source: author)

Phase 2: Selecting Tool for Investigation

Before downloading of the infection or an infected malware specimen can begin, it is important to install all appropriate tools for analysis purposes prior to the system becoming corrupted. To select the tools, free utilities available to the author from various sources have been downloaded and installed; these will be discussed in the next chapter (Chapter 7) in terms of

observing Windows malware. Authentic sources were selected from which the tools were downloaded directly from official websites. The selection of tools was based on a market analysis, which was discussed in Chapter 4, from which selections were made for testing purposes. Some of the selected Toolkit tools are explained in detail.

SmartSniff Packet Analyser

SmartSniff Packet Analyser is a packet analyser tool that works on sniffing network packets that are passing through the network adapter; it also checks the data conversations occurring between the server and client. The complete format can be viewed in both ASCII and test mode for protocols like FTP, HTTP and SMTP or hex dumps (Nirsoft, 2015). Some of the features that were considered vital in the testing are as follows (Black, 2015):

a) **Display Interface:** This tool comes in three different interface modes to view captured data: hex dumps, ASCII and hybrid mode (automatic). In hybrid mode, the tool checks the first stream of data packets and displays data in hex mode or ASCII mode, depending on the number of characters;
b) **Exporting the Result:** The captured result can then be transferred into another application based on the requirements of the investigator/user. By selecting options, the captured data can be copied into a clipboard and then later exported to any supported extensions, such as Excel or Word, or even saved in HTML or XML. Alternatively, a user can select TCP/IP streams and copy/paste them directly into Notepad, Microsoft Word or any editor. Additionally, these can be directly exported as raw data or HTML files. Any captured documents can be attached with other results in the Report Section of the Toolkit; and
c) **Display Filters:** There are several options for filtering the streaming of data packets to investigate different selections, such as for displaying packets only for TCP port 80 and

for including UDP packets for port 53 (DNS).

Buster Sandbox Analyser

A sandbox analyser is a malware detection tool designed for observing the behaviour of the processes and modifications that are performed on the system, and then to undertake an evaluation to detect the presence of malware in the system. The changes can be categorised into various types, such as file system, modification to the registry files and port changes. The following points illustrate some of the important aspects of an investigation using a sandbox analyser (Buster, 2013; McRee, 2012):

a) To obtain valuable information, it is essential to analyse the file system while creating, deleting and modifying. The extension of the file and the section where it was created are the most important factors for collecting valuable information;
b) A sandbox analyser also provides information on the modification performed on the Windows registry. These changes include modulation in value keys, a newly added registry or registries deleted from the system;
c) It provides information on the port changes, especially when a connection is established from outside the system or a port is left open on a local domain. It also collects information when an open port is listening to packets for incoming connections;
d) This tool illustrates a list of the modifications performed by the malware on the system, for example, if any unauthorised software was installed and where exactly it is located; and
e) In addition, it also shows system changes initiated by the actions of the installed malware or infectious files in the system. Issues such as keyboard logging, Window session termination, loading of unwanted drivers and the connection to the internet can

be scrutinised, and then a user can take appropriate actions.

Process Hacker

Process Hacker is a sophisticated tool for monitoring Windows that assists in analysing registries, process trees and threads running on a real-time system. It is an open source tool that views all running processes and includes a memory editor. It also has powerful features, such as termination processes and a Regex memory searcher. One of the main features is its ability to detect malware while troubleshooting for other unwanted activities. Some of the silent features include (Smith, 2010; Source Forge, 2015):

a) Assisting in accumulating more data for investigative operations using input and output parameters;
b) Providing non-destructive features that permit an investigator to filter processes without losing the integrity of the data;
c) Capturing complex stacks of the process threads for each operational activity and making it possible for users to reach the root cause of the problem;
d) Showing each detail of the process activities, including the directory path, the user involved and the sessions identification created on real-time systems;
e) Generating filters for datasets, including the headers, which are not configured as columns;
f) Using advanced architecture for logging activities and scanning up to millions of events in a few minutes and gigabytes of data from logged information;
g) Allowing any investigation to be cancelled at any time during scanning;
h) Capturing records of the processes running from the boot time of the system, making it a more reliable medium of investigation;

i) While scanning a process tree, illustrating all of the associated process trees, which are referenced in a specific pattern and format;

j) Recording native logs syntax, which protects data from being deployed completely over a process hacker tool, making this a persistent tool for investigation; and

k) A user-friendly GUI allowing users to view scanned data in a convenient format, even when it does not fit into the correct column of the result.

The Process Hacker Tool in Artefact (Smith 2010; Source Forge, 2015)

a) Process Tree View: To detect malware, it is important to check all process trees associated with a single executable file. A running process can span into new processes with a parent–child relationship and can be checked in the tree view selection. Each process detected contains at least one thread in which a thread is using the CPU either for actual use or malicious use;

b) Look for Packed Processes: It is common to have packed executable files; alternatively, there can be an encrypted process. Thus, before starting the process hacker tool, it is important to use third-party tools for unpacking files, which then use the process hacker tool for detection;

c) Using Terminator: Sometimes, after detection, it becomes impossible to kill the process. In this situation, using Terminator by right clicking can disrupt the running processes. Sometimes, an investigator needs to manually suspend all running member processes to terminate the tool completely;

d) Looking for Unexpected Sessions: Sometimes malware is hidden in a new session, which can be checked through the Sysinternals Process Monitor Tree. Session 0 and Session 1 are normal for any system, but having additional sessions could be a sign of malware in the session; and

e) Looking in Service.exe Files: Through process hacker, all running services can be

viewed through sc.exe. There could be some non-running and non-interactive services running on the system. Using the sc.exe built-in tool can help with interacting with the service.exe and detecting anomalous files.

Wireshark

Wireshark is a packet analyser tool that captures network packets, showing the details of the network to the user. It is similar to a measuring device to check the on-going activity of the network in detail. For many investigations or the detection of malware activities, Wireshark can often be regarded as the best tool to use. Some of the most important features of Wireshark are (Cheok, 2014; Kurose, 2009):

a) The tool is available and can run comfortably on both Unix and Windows platforms, providing flexibility for investigators;
b) It uses a network interface to capture live data feeds present in the network;
c) The *open* option provides detailing of the files and other data packets that were captured with *tcpdump* or any other packet capture tool available;
d) It displays data to users with all of the detailed information on the protocols captured;
e) It can import and export data packets in other file formats to simplify documentation for the user;
f) Users can save the captured data packets for future use and analysis;
g) The search button can assist an investigator in filtering out captured packets to resolve the intended problem or scenario; and
h) Investigators can convert the captured data into statistical data for further explanations and analyses.

Using Wireshark in the Artefact for Malware Detection (Cheok, 2014; Kurose, 2009)

a) **Changing the Display of the Column in Wireshark:** Wireshark can be initiated by changing the display button of the column, as the display button does not work properly for investigative purposes, especially in the case of detecting malware in the network;
b) **Filtering Protocols:** Use the 'filter' option to search for a particular protocol used in a testing artefact;
c) **Monitoring Website Access:** This can be a good source to detect malware, carried out by breaking traffic into protocol differentiation. Typing 'http' into the filter box can assist in viewing only traffic coming from http;
d) **Endpoint Dialog Option:** Using the Endpoint dialog, the list of the websites being accessed on the system, websites from third parties, unknown sources and advertising networks can be viewed in a separate box to check for abnormalities; and
e) **Plug-in Detection:** Noticing and detecting plugins can be very useful in detecting malware in the associated system. Plugins can be vulnerable, as they can attack browsers. To detect the plugins in systems, Plugin Detect can be used by including 'http contains stat' in the filter option.

6.4 Risks and Issue Analysis

Considering the artefact as a prototype for the proposed framework, issues and risks are involved in the development. These are briefly outlined as follows:

a) Initially, a lack of expertise in the programming language (.NET) was problematic. Learning and enhancing capability from various sources, including university resources, helped in optimising the development for later stage;
b) The Toolkit is an amalgamation of open source tools and requires learning, training and tutorials for performing investigations and associated experiments;

c) Every experiment requires a sample malware to be downloaded from sources for testing and experiments, which creates a risk of infection in the system. Thus, substantial time was invested in developing a safe environment, preventing the master machine from loss of integrity.

6.5 Summary

The intention in this chapter was to develop an artefact that reflects the critical functionalities of the proposed framework mentioned in Chapter 4. The artefact is in the form of a malware analysis toolkit, which is simple in structure, yet provides a robust system that integrates open source malware detection software tools. The developed Toolkit provides a flexible skillset to critically respond to current cyber incidents in any dynamic forensic environment. Furthermore, due to the Toolkit allowing further code and modifications to easily take place, future cyber incidents will also be responded to as malware itself changes as future trends and developments occur. It will support an investigator in analysing given malware on complaint management, and ensure security during the investigation through robust authentication and authorisation methods. It will perform a rapid request and response on the results while creating effective reports on the findings for further analysis. It also aimed to provide a reasonable and useable standard upon which law enforcement could act accordingly. Although this was something that could not be directly measured without conducting tests within a law enforcement environment and would need to be part of future development, it was an important consideration for the longevity of the final product. For developing the software within the respective domain and timeframe, a Waterfall model of the software lifecycle was selected as a standard model for development. This comprised four phases: requirements and gathering; design; implementation; and testing. In the requirement phase, the focus was on listing functional and non-functional requirements of the system, which included lists of required hardware and software. For developing the front-end design of the system, .NET was used as a

programming language while MySQL was used for developing the back-end services. After implementation was complete, testing was initiated by executing and running the system to observe its performance. Before initialising this phase, it was important to create suitable environmental conditions for running the system. Thus, a safe environment was also planned in accordance with testing and observation. Before downloading of the infection or an infected malware specimen could begin, appropriate tools were installed for analysis purposes prior to the system becoming corrupted. To select the tools, free utilities available to the author from various sources were downloaded and installed.

The use of free utilities for this research was a purposeful decision, which was considered to have little effect on the outcome of the overall results. Many different commercially available, freeware, and open source tools are available, but performance is not directly linked to the upfront costs of the tool (Rastogi, 2013). Although some tools have been shown to outperform others (AV Comparatives, 2013), with frequent changes to malware keeping pace with updates to malware tools (Malwarebytes, 2017), it is not necessarily the appropriate approach to select a more expensive commercially available tool. However, this can be a consideration for further work in the development of a full-scale version of the software.

CHAPTER 7 - RESULTS AND OBSERVATIONS

7.1 Overview

Testing of the Toolkit was based on the work discussed in the previous chapter (Chapter 6); the required hardware and software were obtained and utilised. From creating the working environment (an isolated lab) to the testing environment, each phase was important for testing. This chapter illustrates the testing attributes used to deploy a successful Toolkit functionality. Moreover, it presents the theoretical and practical implications faced during the implementation and testing phases. The choice of environment and functional elements for implementation were dependent on the problem statement, as well as the market analysis performed, as described in Chapter 6. A thorough evaluation was undertaken; thus, it was useful for the author to explain the needs and requirements for the Toolkit. This activity helped the author to enhance the performance of malware detection and analysis more robustly and efficiently, and assisted in achieving better investigatory results. The Toolkit is an amalgamation of various successful pre-existing open source tools, thereby eliminating the need to develop a completely new tool for malware detection. Thus, the implementation focused on a user-friendly solution to make the complete process of investigation usable and scalable in any dynamic environment.

The Toolkit is a novel approach designed to create a better investigative environment and robustness in malware detection. As mentioned in Chapter 6, this Toolkit is a prototype that focuses on vital attributes, such as reducing the number of investigative cycles, providing a safe and controlled environment for investigation, performing both static and dynamic investigations through proactive open source tools, allowing for an efficient incident response in situational analysis and using a single platform for multiple malware analysis. Highlights of this tool include the removal of irrelevant tools, the saving of investigated data, and by minimising the amount of tools used in an investigation, there are also possible cost-saving implications.

As mentioned previously (Chapter 6), the development of the Toolkit was programmed using the .NET language for developing the front-end interface, while MySQL was deployed for back-end database development in the system. Before implementation, there was a need to ensure that the development and execution would occur within a safe and controlled environment. To accomplish this, the concept of a virtualised operating system was used. VirtualBox software was installed with a Windows ISO image file to protect the actual network system from any infection or any associated risk during the experiment.

7.2 Environmental Setup

To investigate and test the Toolkit, it was essential to create and maintain a safe environment that consisted of a virtual system placed on the actual system. The scope of the environment was not only restricted to the software, that is, the operating system, but also to the network resources and other related services. In this scenario, malware is modern in nature and directed towards networks and other running services. Thus, it may become necessary to emulate the environment, helping malware to run completely for testing purposes and used for common services, such as the domain name service (DNS) or simple mail transfer protocol (SMTP). To create this environment, VirtualBox by Oracle, VMware or Virtual PC could have been used. For the present development, the author selected VirtualBox to perform test experiments to create a safe environment. This was selected due to the author having familiarity with the software, although any of the aforementioned software could have been chosen based on performance and suitability. To give an appreciation of Oracle VirtualBox environment, selected screenshots have been provided in Figure 19 below:

Figure 49: Selected Screenshots of the VirtualBox Environment (Source: author)

After the entire source code was created and run, the application was transferred to a safe environment for testing purposes. Before the Toolkit could be tested, it was important to load malware samples in a reserved, specific folder, which was later used for results and analysis. This virtual environment was beneficial as testing in a physical environment would have been time-consuming, and the ability to restart the machine in any failure situation allowed the experiment to be saved. This clarified the researcher's purpose of distinguishing the system's infected and uninfected zones. In addition, there were numerous precautions taken during the experiments. For instance, in scenarios in which malware was used, it could have acted differently if a virtual environment in the system were detected. Other issues arose in which the malware tried to trespass the virtual system to enter the base system; however, VirtualBox provided a useful tool in resolving these issues (Table 12).

Issue	Positive	Negative	Comments
System performance before infection	✓		System performance was monitored and kept running to check its smoothness
Malware infection	✓		Sample malware execution files were deployed successfully for the experiment
Malware penetration in the virtual system	✓		The virtual system responded in a positive way, preventing malware from trespassing through the virtual system and safeguarding network assets
System performance after infection		✓	It was difficult to run the prototype due to the high infection rate in the system from the sample malware

Table 12: Issues Faced and Prevented During the Environmental Setup for the Toolkit

7.3 Working Features of Toolkit

A step-by-step explanation of the features and functionality of the Toolkit can be provided by illustrating the user interface of the system. Some of the screenshots are shown in detail in Appendix C.

Running the Toolkit begins with a Login Page comprising an authentication process in the form of ID and password. Once the user is authentic on the basis of the access rights, he will be authorised to use the Toolkit. Some features after successful login are described as follows:

a) **Login Panel:** This includes two user profiles, User Login and Admin Login. Authentication is described based on the rights given to the user type. The Super Admin

can login through a default name and password, which will be created during development, and new users (investigators) can be added later by the Super Admin on a rights basis. Functions in report creations, such as read/write, can be used by the Super Admin, while only the read option is given to the user. This logs the users on, and their activity can be viewed by the Super Admin for analysis;

b) **Scan**: This functionality will allow admin to start any investigation by selecting any particular category and respective tool to start the execution of the system. Once the software is executed and starts running, a new panel for capturing and recording the screen result will pop up. User/investigator can record the result at any time of investigation. Scanning cannot start without adding a title for the investigation. Captured screenshots will be automatically added to the investigation;

c) **Report**: Once the scanning is done, a user can then create its own standard report. The report section will allow any user to view/edit reports. Edit will allow users to add/delete/modify the description and keywords with respective investigation (scanning). Super-admin can view/add/delete any user's report while other users can view reports on the basis of access control;

d) **Manage Software:** This functionality is only allowed to the Super Admin. Through this functionality, users can add/delete/edit any tool category and respective software. Users can also add/delete/modify the path for database storage; and

e) **Manage User**: This functionality is only allowed to the Super Admin. Through this function, users can add any investigator or end user in the list of authentication. They can also activate or deactivate the respective user.

7.4 Login as Super Admin

After successful login to the Super Admin section using default passwords (which can be changed later), a user is then able to see a user interface, giving access to all the respective

functionalities mentioned above. It will show different sections (headers) at the top of the panel as shown in Figure 50.

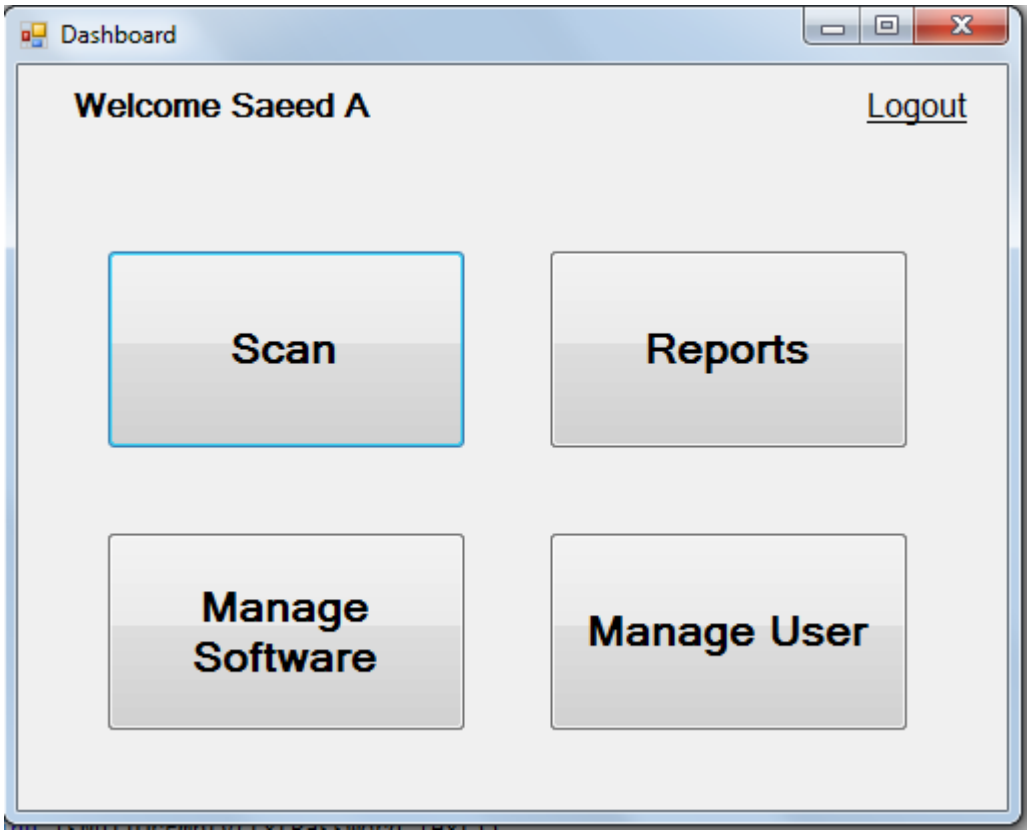


Figure 50: Screenshot of the Dashboard after Super Admin Login (Source: author)

7.5 Collection of the Sample Malware

Malware samples can be taken from various sources. Malware is available online on certain websites and samples can also be taken from educational institutes carrying authentication to provide the malware samples for academic purposes. The authenticity and validation of the sample malware was tested online using MD5 checksum, which although has vulnerabilities, is adequate for verifying data integrity against unintentional corruption (NASA, 2015). Some of the used online databases include popular websites, such as Virus Total and Malware Hash

Registry. It was important to complete all of the environmental containments before downloading or transferring any collections onto a system.

To implement the abovementioned process, it was vital to design a framework to develop a toolkit for detecting and analysing malware files and network resources, acting as a prototype for digital forensics and security. The Toolkit is capable of carrying out processes that can assist in analysing the infected files and checking the optimisation of open source tools available worldwide.

7.5.1 Obtaining and Installing the Sample Malware

A process was developed based on work outlined by Zeltser (2015), which provided guidelines on how to safely install a malware sample for testing purposes.

Running the prototype was only initialised when a working environment was ready with a proactive malware sample installed in it. Selecting sample malware was important for the testing environment. Sample malware was downloaded from trusted websites and other sources following ethical considerations. These sources carried malware samples intended only for research and investigative purposes. The following factors based on known best practices were considered before starting the investigation:

a) Before downloading, it was vital to ensure the integrity of the virtual environment (safety) and to prevent spreading the infection to the base system or any other wired or wireless network nodes;
b) Special consideration was taken to understand the nature of malware and the scope of infection, which assisted in selecting the relevant tool in the Toolkit for investigation;
c) Special folders and sub-folders were created and marked to remember the location of the installed file;

d) After installation, processes were checked and monitored continuously to store knowledge and data on behaviour, which could later assist in determining the success rate of testing the desired tool in the Toolkit;

e) The status of the system was recorded for comparison at a later stage with the status after the infection;

f) Special consideration was made to turn off firewalls, anti-virus software, Windows Defender and any other commercial tool already installed on the system. This was done to prevent such programs from having any implications later during malware deployment and testing; and

g) Network statistics, such as the IP address and subnet mask of the system, were recorded to distinguish between authorised and unauthorised sources in the experimental results.

7.6 Testing the Prototype

This prototype was developed with the objective of illustrating both theoretical and practical implications. The implementation is still in a raw stage and inevitably would need refinement and further testing prior to a finished product being finalised. However, this section illustrates and explains the testing done to date for the Toolkit, with descriptions of selected experiments carried out on certain tools with an analysis of raw malware sample files. Source coding of the Toolkit, sources of the malware and other testing screenshots are located in Appendix E and Appendix C.

Testing the tool has its own importance. Testing was required to ensure correct and adequate operation was achievable. To accomplish this, three research experiments were conducted selecting the following tools: Process Hacker, SmartSniff and Wireshark. Considering any limitations (outlined in Section 6.7), the experiments were performed successfully, preventing

any associated bottlenecks in terms of performance, flexibility, learning and adaptability of the system. Before testing of the system could begin, it was important to confirm the availability of a controlled environment detached from the existing network, also known as a base system. Once this virtualised environment was ready, the malware was installed. All installed folders were checked for appropriate malware and were run for the selected tool. The above process was followed and repeated for all respective tool categories. Selected screenshots for the experiments are shown from Figure 20 to Figure 30.

7.7 Legal and Ethical Considerations

It is important to consider legal and ethical issues during a malware investigation. From the perspective of legality, it is vital to ensure correct handling of malware samples, their preservation within the system, collecting evidence and presenting this evidence in safe custody. It also includes disclosure of private data (if any) recovered during the investigative operation. This private data may include IDs, passwords, personal credentials, such as phone numbers and email addresses, and financial data, such as credit and debit card information.

Following ethical considerations confirms that the malware process has been undertaken in a secure and realistic manner and that all possible preventative steps have been taken to prevent stored malware from spreading. In addition, in a situation in which an investigator may find a new piece of malware, sharing the knowledge of the finding also falls under ethical considerations. Guidelines on ethical considerations are outlined in Dittrich and Bailey (2013) and followed as practically as possible for the completion of this work.

7.8 Experiments on the Toolkit

Prior to the experiments, it is important to structure the actual inputs and the desired/expected outcomes. The inputs that were used during the study involved the use of malware samples with their distinctive characteristics, mentioned as follows:

Table 13: The Sample Malware as Input with Respective Attributes

Sample Malware (Input)	**Attributes of Malware after Infection (Output)**	**Tool Used for Detection and Analysis**
Stuxnet (Worm) Tapas_6300.exe	• Infected executable • Irrelevant CPU usage • Degrading system performance • Parasite component • Payloads	Process Hacker
Spyware Package	• Unwanted usage of Port 80 • Major unauthorised changes and sending system activities outside the network • Directly sending messages through unwanted TCP and UDP transmission • Using network resources • Anomalous packets on the system	SmartSniff
win32/polip.A	• Three-way handshake with unwanted destinations • Unrecognised peer connection with external network • Malformed traffic from various unauthorised sources to the system	Wireshark

The following section details a series of experiments that were carried out on the Toolkit and the observations made by the author based on the performance and outcome. The experiments

were selected based on choosing different but potentially high-risk malware types to rigorously test the Toolkit and show its effectiveness in real-world situations by using actual malware.

Experiment 1

Name of the Packaged Malware: Stuxnet (computer worm), Tapas_6300.exe
Type of infection: Infected executable files, irrelevant CPU consumption, performance degradation
Targeted Category: Process monitoring
Tool Used: Process Hacker

Stuxnet malware is a packaged computer program that was installed intentionally with prior knowledge of its variable infections. Through this test, the system was scrutinised and recorded in the form of screenshots that were saved in a specific folder (as selected by Admin). Relevant screenshots related to the result are presented in

Figure 51.

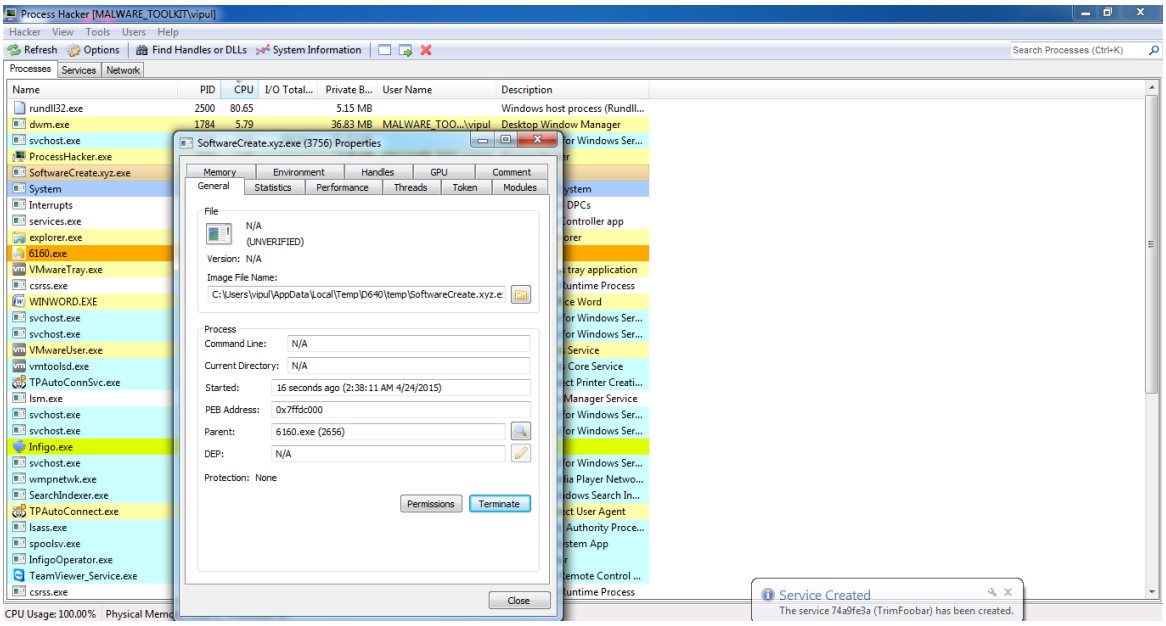


Figure 51 Screenshot Showing Infected Executable Files for Experiment 1 (Source: author)

Observation 1: Various infected files can be seen in Figure 21. For instance, an executable file called ‘SoftwareCreate.xyz.exe’ was detected, which could be terminated if required by the investigator. According to white papers on spyware (Dizon et al., 2010), any executable file with a xyz.exe format initiates malicious processes, launching parasite components and payloads that are destructive in nature (Dizon et al., 2010). The presence of this file showed that the system was injected with a vicious threat. Files such as xyz.exe run silently in the background and can only be checked by investigators through tools such as Process Hacker. These files are difficult to delete by any anti-virus or anti-malware software; thus, manual deletion by an investigator is required to prevent any further loss. In Observation 1, the aim of the experiment was to capture the Protocol Hierarchy Statistics, which illustrate the total data usage by a particular IP address and the interaction of the system with unrecognised sources.

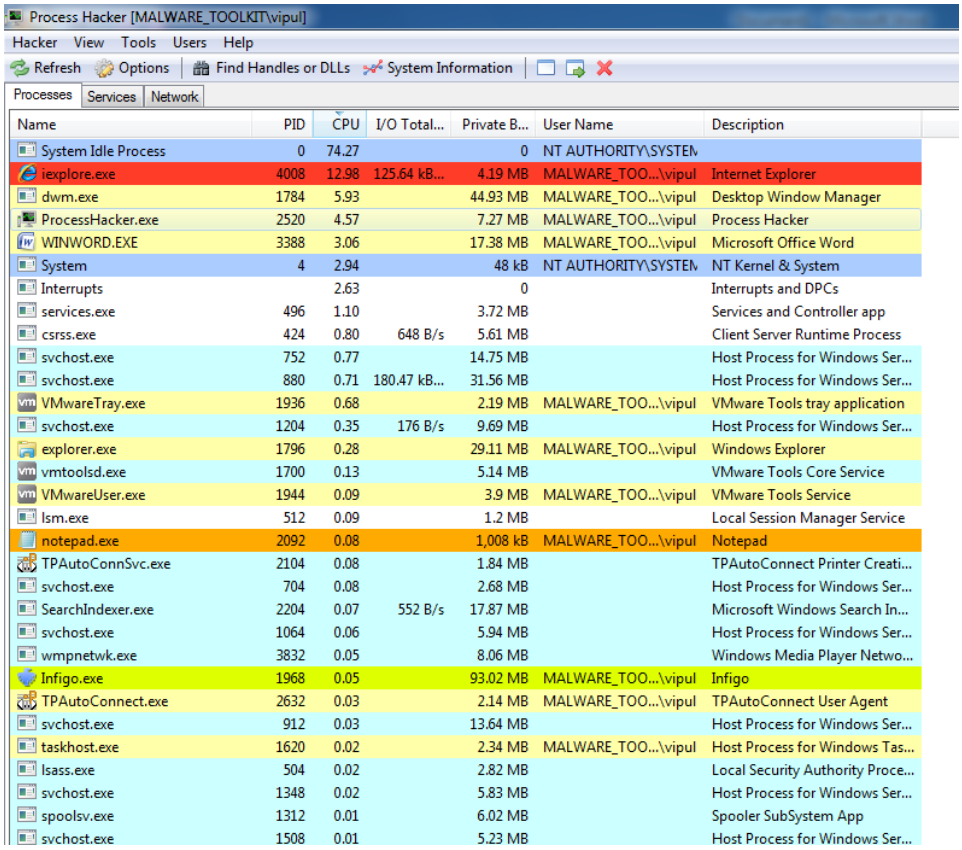


Figure 52: Screenshot of Injection on an Explorer.exe File in Experiment 1 (Source: author)

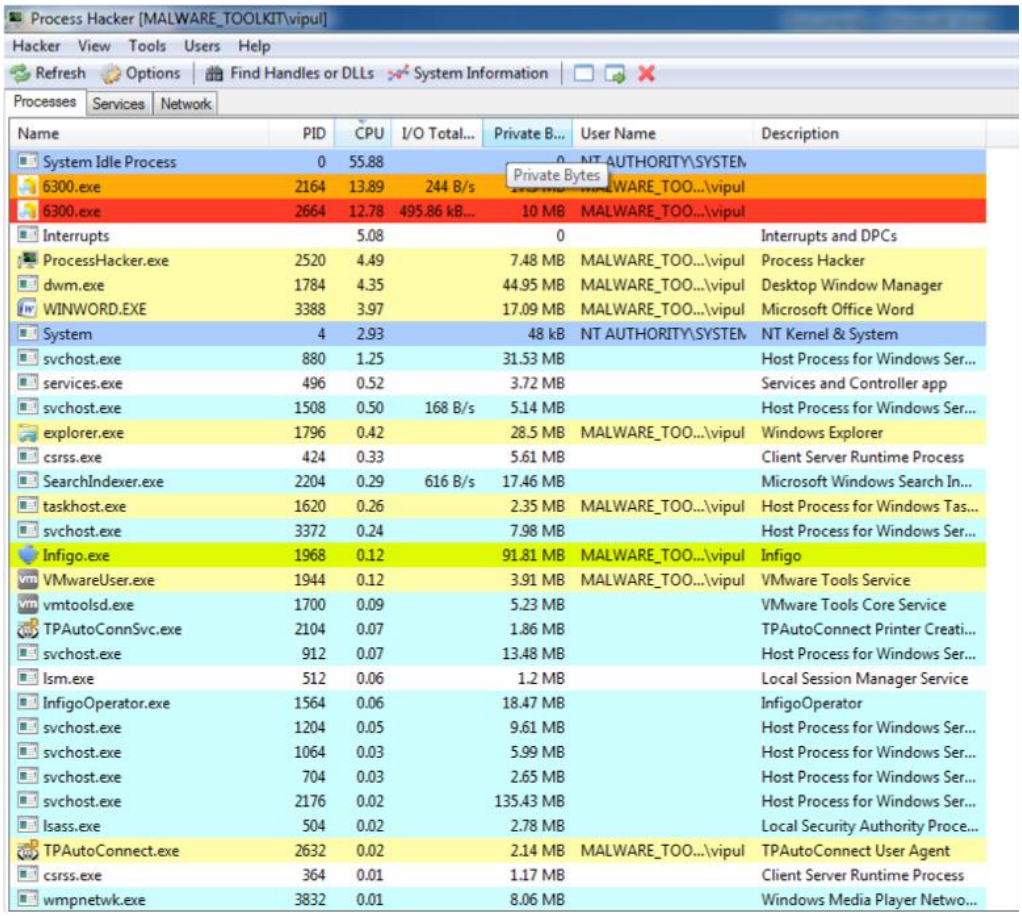
Observation 2: Figure 22 shows a red line highlighting iexplorer.exe. If running in the background, iexplorer.exe shows symptoms related to a virus or malware injected on Microsoft Explorer. Although not critical to the health of the computer system, it can activate multiple explorers at the same time using CPU processes and bandwidth, causing delays for the user. The common problems and symptoms of this infection are as follows:

a) Slow computer;

b) Slow internet connection or working of Internet Explorer;

c) Processes like ‘svhost’ consuming 100% of CPU; and

d) Browser randomly crashing.



Name	PID	CPU	I/O Total...	Private B...	User Name	Description
System Idle Process	0	55.88		0	NT AUTHORITY\SYSTEM	
6300.exe	2164	13.89	244 B/s	[illegible]	MALWARE_TOO...\vipul	
6300.exe	2664	12.78	495.86 kB...	10 MB	MALWARE_TOO...\vipul	
Interrupts		5.08		0		Interrupts and DPCs
ProcessHacker.exe	2520	4.49		7.48 MB	MALWARE_TOO...\vipul	Process Hacker
dwm.exe	1784	4.35		44.95 MB	MALWARE_TOO...\vipul	Desktop Window Manager
WINWORD.EXE	3388	3.97		17.09 MB	MALWARE_TOO...\vipul	Microsoft Office Word
System	4	2.93		48 kB	NT AUTHORITY\SYSTEM	NT Kernel & System
svchost.exe	880	1.25		31.53 MB		Host Process for Windows Ser...
services.exe	496	0.52		3.72 MB		Services and Controller app
svchost.exe	1508	0.50	168 B/s	5.14 MB		Host Process for Windows Ser...
explorer.exe	1796	0.42		28.5 MB	MALWARE_TOO...\vipul	Windows Explorer
csrss.exe	424	0.33		5.61 MB		Client Server Runtime Process
SearchIndexer.exe	2204	0.29	616 B/s	17.46 MB		Microsoft Windows Search In...
taskhost.exe	1620	0.26		2.35 MB	MALWARE_TOO...\vipul	Host Process for Windows Tas...
svchost.exe	3372	0.24		7.98 MB		Host Process for Windows Ser...
Infigo.exe	1968	0.12		91.81 MB	MALWARE_TOO...\vipul	Infigo
VMwareUser.exe	1944	0.12		3.91 MB	MALWARE_TOO...\vipul	VMware Tools Service
vmtoolsd.exe	1700	0.09		5.23 MB		VMware Tools Core Service
TPAutoConnSvc.exe	2104	0.07		1.86 MB		TPAutoConnect Printer Creati...
svchost.exe	912	0.07		13.48 MB		Host Process for Windows Ser...
lsm.exe	512	0.06		1.2 MB		Local Session Manager Service
InfigoOperator.exe	1564	0.06		18.47 MB		InfigoOperator
svchost.exe	1204	0.05		9.61 MB		Host Process for Windows Ser...
svchost.exe	1064	0.03		5.99 MB		Host Process for Windows Ser...
svchost.exe	704	0.03		2.65 MB		Host Process for Windows Ser...
svchost.exe	2176	0.02		135.43 MB		Host Process for Windows Ser...
lsass.exe	504	0.02		2.78 MB		Local Security Authority Proce...
TPAutoConnect.exe	2632	0.02		2.14 MB	MALWARE_TOO...\vipul	TPAutoConnect User Agent
csrss.exe	364	0.01		1.17 MB		Client Server Runtime Process
wmpnetwk.exe	3832	0.01		8.06 MB		Windows Media Player Netwo...

Figure 53: Result of Infection through Executable File 6300.exe in Experiment 1 (Source: author)

Observation 3: The Process Hacker Toolkit screenshot in Figure 53 shows an infection by a 6300.exe file. This is ransomware that can encrypt files on a system without prior permission. Once this program infects a system, it can lock several other programs on the infected computer and even if an investigator tries to restore files, he or she may still see files with a 6300.exe extension. Ransomware prevents users from accessing files on their system, typically until a ransom payment is made, hence the term ransomware. As a computer is essentially locked from the user and is of limited or no use, users may see no way of regaining access and succumb to the demands of the malware request for payment.

Experiment 2

Target Category: Network analysis
Target Malware: Suspicious malicious network traffic
Tool Used: SmartSniff (open source)

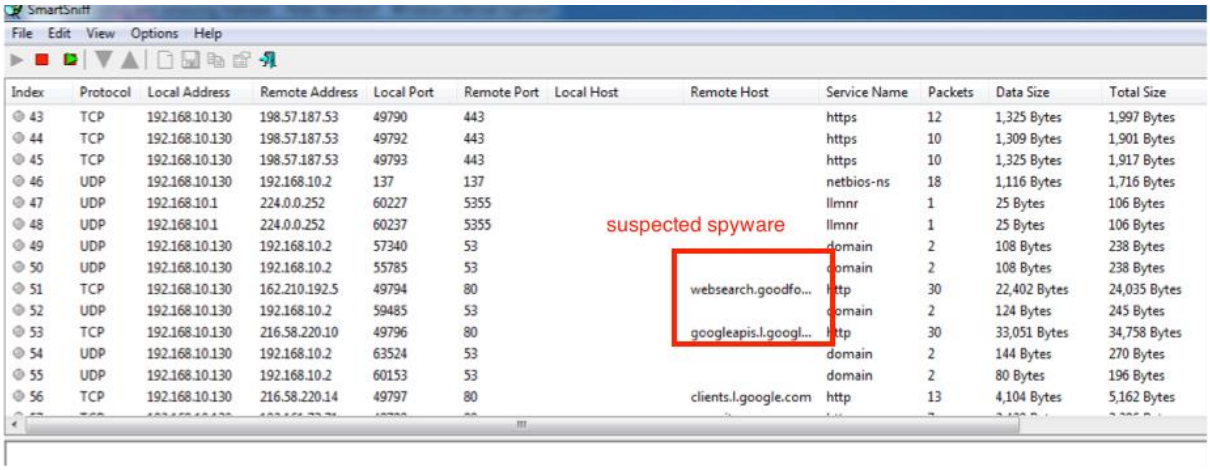


Figure 54: Results for SmartSniff after Sniffing Network Traffic in Experiment 2 (Source: author)

Observation 1: Figure 54 shows some of the suspected traffic from spyware. For instance, websearch.goodforsearch.info is a hijacker that harms browsers. Once installed in a system, this malicious tool can harm the victimised system. The presence of this malicious spyware overtakes the current browser and replaces the homepage with websearch.goodforsearch.info. This activity can harm the system by making major unauthorised changes and sending system activities by directly sending messages through unwanted TCP and UDP transmissions. Attackers can easily manipulate systems by downloading and installing malware tools or programs and block users from accessing websites. This malware activity can flood a system with adware and popups resulting in more CPU usage and slowing the computer system. This tool has provided evidence to show the installation of websearch.goodforsearch.info malware

and further action can then be taken to clean the infected system, highlighting the benefit and need for detection.

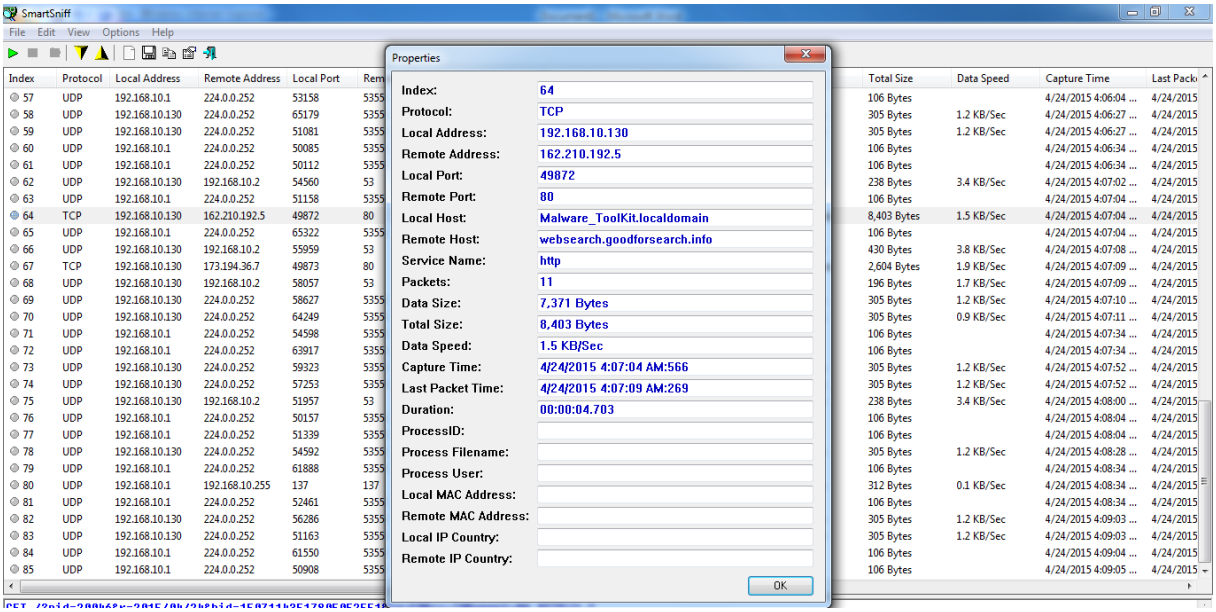


Figure 55: The TCP Stream for Captured Packets in Experiment 2 (Source: author)

Observation 2: By clicking on the TCP stream option in the tool or by double clicking, another panel appears, presenting more information as shown in Figure 55. It focuses on giving information on the protocol, its remote address and remote host information; this is vital information for an investigation. Here, the remote address is 162.210.192.5, which can be searched on IP tracer tools to obtain detailed information related to the geographical location of the hijacker.

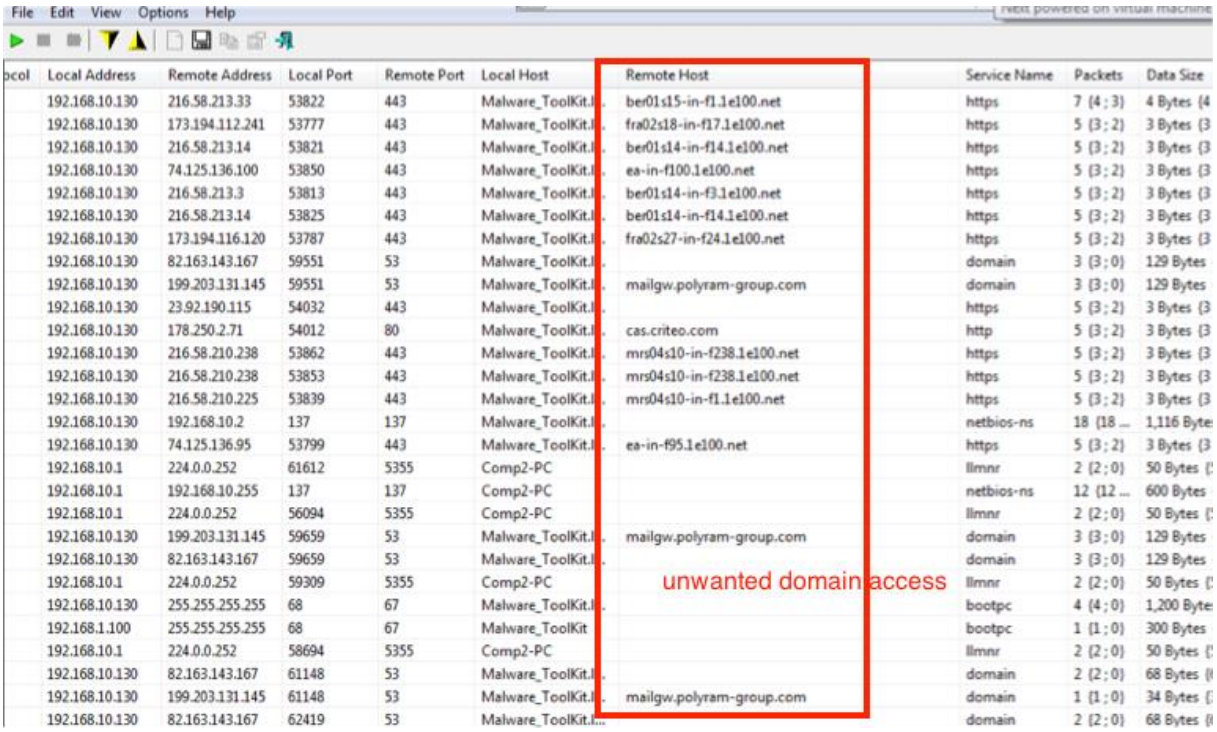


Figure 56: Another Search for Suspected Unwanted Domain Access in Experiment 2 (Source: author)

Observation 3: In this observation, the author found some of the changes in the system as shown in Figure 56. After the malware was successfully installed and spyware was detected, unwanted malicious traffic was detected by the SmartSniff tool; it was found to be turning the browser activity off completely. Focussing on the captured remote host information, there were multiple websites being accessed on UDP port 53 (DNS), resolving the IP address to the domain and vice versa, and sending messages to the servers without prior permission. Moreover, TCP port 443 (http) unsecured access was also observed. These statistics and captured anomalous packets show the presence of malware in the system. On further investigation, the author selected one IP address, 199.203.131.145, on an IP tracer, and found links to a server located in Israel without the website being accessed from the browser. This tool again highlights the importance of detection where unwanted, and possibly malicious, activity is

being forced upon a computer system without the user's knowledge or intention. Identification is the first step before required cleaning takes place, and without this identification, unwanted activity would continue.

Experiment 3

Target Category: Network analysis
Target Malware: win32/polip.A
Target Tool: Wireshark

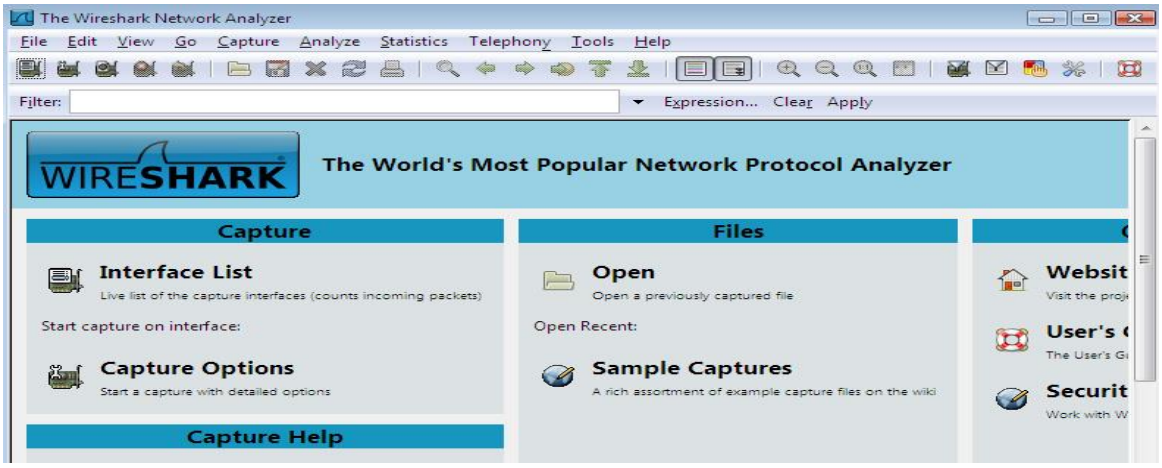


Figure 57: Starting the Wireshark Tool in Experiment 3 (Source: author)

The aim of this experiment was to exploit the presence of unwanted traffic patterns to the local node; that is, to track all packets from the external nodes, showing unwanted access to the system in a real-time scenario (RTS). In this experiment, there were two main nodes, of which the Toolkit was working on 192.168.10.129 along with Netmask 255.0.0.0. The target malware was downloaded and run for the experiment. Figure 57 shows the start-up screen of the Wireshark tool. The system was enhanced by a public network with anti-virus detection tools

deactivated to ensure any unwanted access would be granted to the system and ultimately demonstrate the effects. The following figures are some of the captured screenshots illustrating the anomalous behaviour in both incoming and outgoing streaming.

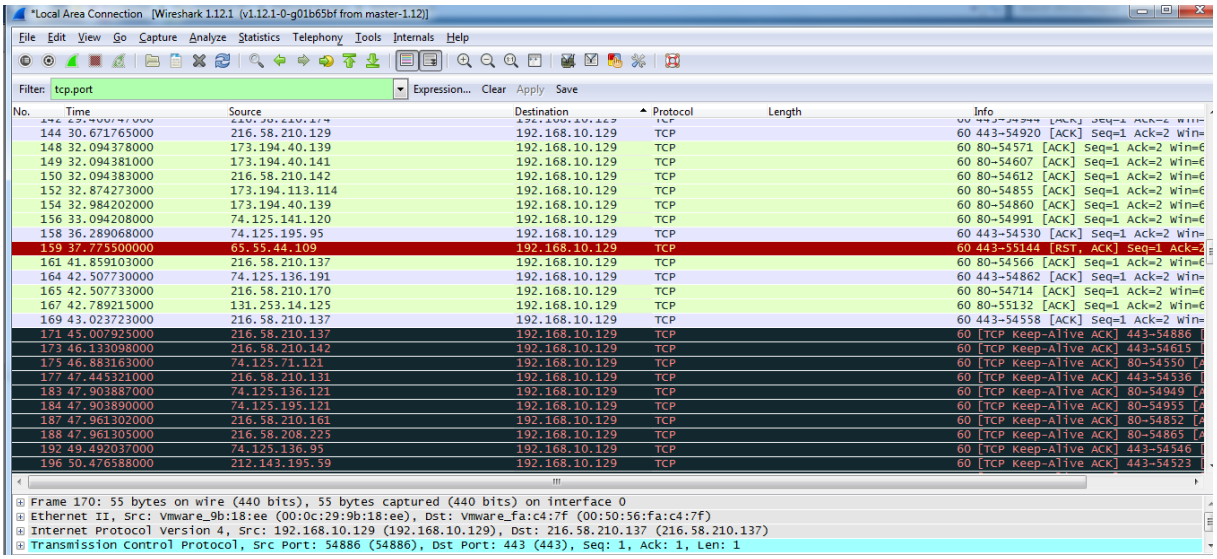


Figure 58: Screenshot of Results with the Captured Packets from Wireshark in Experiment 3 (Source: author)

Observation 1: The captured packets showed the presence of unwanted packets from unwanted sources in Figure 58. At this time, all browser activity was halted. From a user perspective, this meant that the system was not involved in any IP interaction with any other machine. The presence of packets at the TCP port showed a three-way handshake with unwanted destinations. The red-coloured packet illustrates that it carried a serious threat, pointing to the presence of malformed traffic. This three-way handshake provides an opportunity for malicious servers to bypass security scanning on the networked devices and allow for further malicious attempts to occur. The presence of a TCP RTS packet was also observed, which meant the connection on which the previous IP was selected for sending data packets was no longer recognised. Moreover, this showed an ASK in a RST==0, which was attempting to

establish a peer connection that was unrecognised by the system. Such unwanted peer connections could be a source for malware and could host malicious content and, hence, would not be a welcomed system feature.

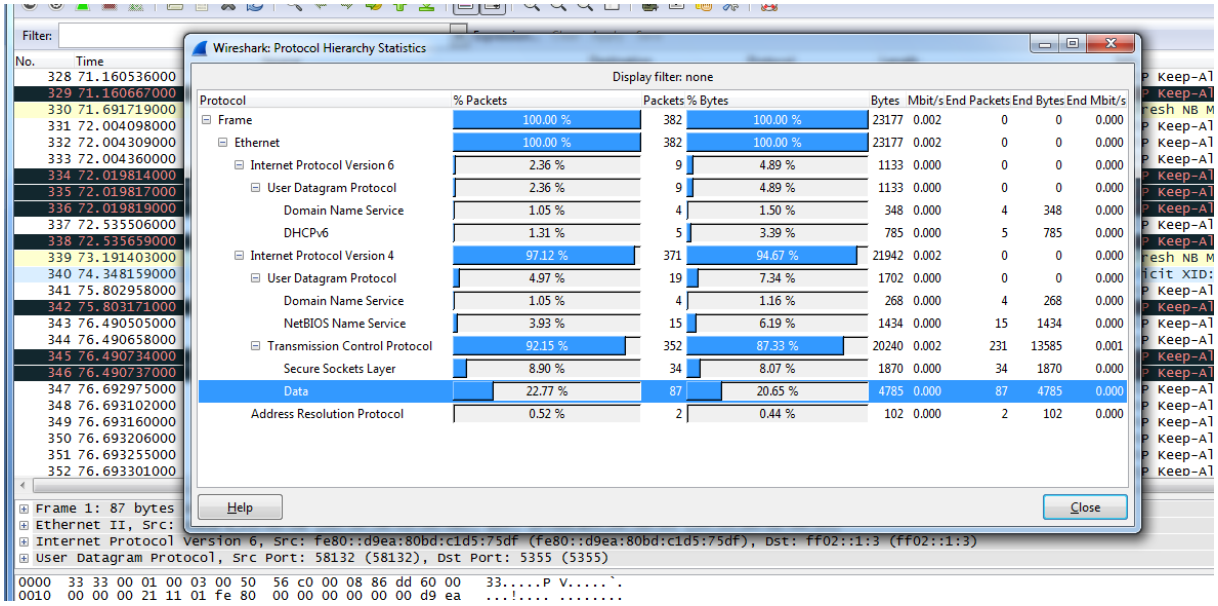


Figure 59: Screenshot Displaying the Protocol Hierarchy in Wireshark in Experiment 3 (Source: author)

Observation 2: The above screenshot (Figure 59) shows the protocol hierarchy for the captured packets in Wireshark. The presence of data (unrecognised packets) under the TCP hierarchy was analysed; statistically, these represented 22.77% of the total packets captured by the tool. This uncategorised data represented the presence of malware in the system, as Wireshark was not able to distinguish any other protocol.

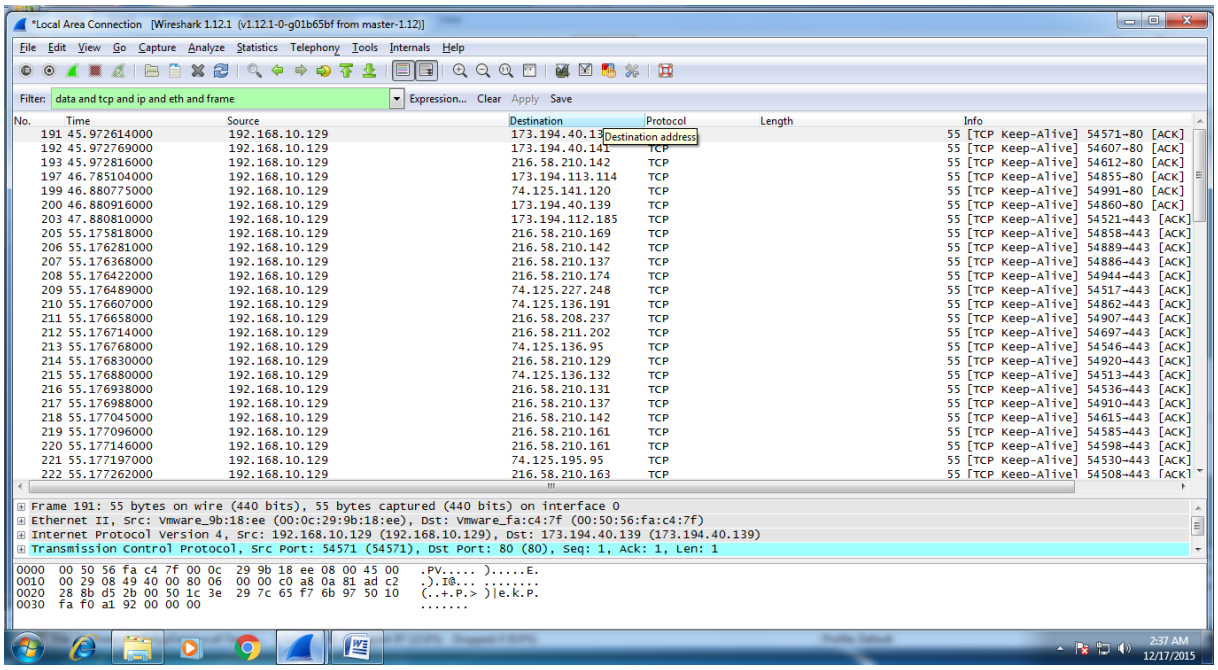


Figure 60 Screenshot of TCP Port Filter in Experiment 3 (Source: author)

The image in Figure 60 prominently shows details on uncategorised data packets under the TCP frame, carrying particular details concerning source and destination. These unrecognised packets could be further analysed based on the TCP stream.

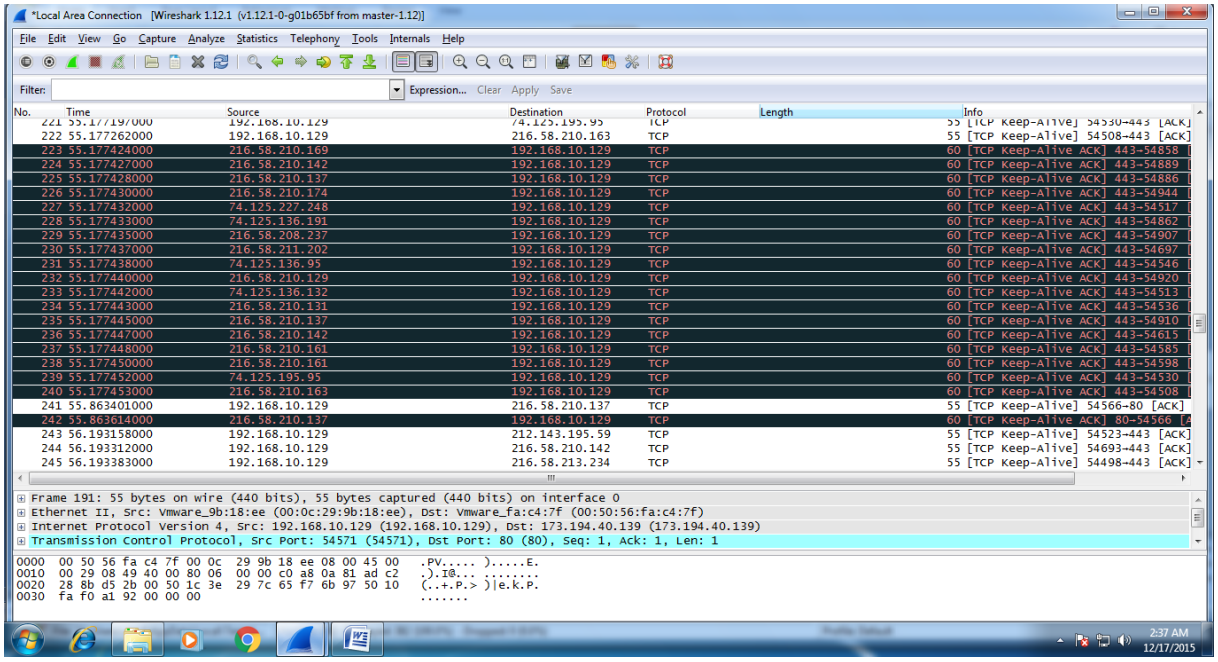


Figure 61: Screenshot Displaying Captured TCP Packets in Experiment 3 (Source: author)

Observation 3: The above screenshot (Figure 61) illustrates the packets in black with red text packets, which indicate bad TCP packets that were trying to connect with outer sources. This is indicative of malformed traffic from various unauthorised sources to the system. The intention of this malformed traffic is to disrupt services on the home network and possibly reveal availability of services to the unauthorised source, thus causing issues and possible harmful repercussions within the home network.

The results observed from all three experiments were matched with the expected outcomes listed prior to the experiment (ref: Table 13), thus validating the results of the experiment.

7.9 Discussion and Implications

This section focuses on implications associated with the Toolkit's implementation and results of the experiments. The objective of deploying the prototype was to provide support for research conclusions to benefit digital forensics and security. The author's intention was to conclude by illustrating facts based on the framework and link these to the selected problem statement.

7.9.1 Theoretical Implications

The tools involved in deployment were selected after extensive market analysis in which questionnaires were used to generate a unified result. The current investigation included manual operations and analysis using selected digital forensics tools. Information gathered during prototype implementation required detailed efforts of scrutinising and formulating an understanding of the conclusions. When presenting the information in a format to ensure understanding, the following were considered: prior knowledge and expertise before defining the input data was vital, as was keeping the artefact under the selected domain; and to help the understanding of how data could be manipulated and paired to form conclusions, pre-processing of data and the use of link mining techniques were assessed.

This research focussed on enhancing knowledge through identifying and detecting traces of malware in an infected system.

7.9.2 Practical Implications

The practical implementations of artefact deployment cover several necessities and aspects of digital forensics, providing solutions for all scenarios in digital security and investigation. None of the experiments conducted by the author were performed in a generic environment or condition. All testing and observations were carried out based on limited available resources at the time, specific configurations and computational power. Moreover, the Toolkit was developed for one-tier systems and is not applicable for larger environments using multi-tier systems. However, its

performance could be enhanced in future works and reconfigured in accordance with the scenario and situation in which it is used. The system requirements, including technical requirements, would need to be changed for it to support a large-scale environment. This method needs to be considered by investigators and security experts regarding how it can be further improved and utilised. While the current version of the Toolkit is purely a desktop-based application, a web-based application can be a future option to eliminate computational and resource limitations. Other options could also be utilised to enhance the computational power, such as using a cloud-based infrastructure, which again would mitigate the risk of resource dependency.

Performing experiments in a virtual environment provided various benefits. However, it also had some negative effects on the results. During the experiments, numerous VMware files were found on the machine in a cluster format and the functionalities of the Toolkit were unable to recognise them, contradicting the test outcome. In every experiment, consideration was given to files possessing VMware cluster formats; such files were ignored and removed from the observations. However, the whole process was manual and, therefore, it was unlikely for it to be 100% reliable. Nevertheless, if malware sensitive to virtual environments was involved, it would be impractical to perform the experiment. In addition, the virtual environment results in clustering different machines together, creating a different environment for malware than a real compromised system, possibly affecting results. To this end, further testing within a real (non-virtual) environment would be ideally sought to remove any of these highlighted concerns, and although this is largely impractical from a research viewpoint due to resources, it would prove its suitability beyond the conditions in this test.

Another implication in the practical experiment was the way in which results were presented. The presentation in this scenario could be different from an actual investigation in a real scenario, where a tactical investigator would then evaluate the analysis. This investigator would

write reports in consultation with a legal practitioner before presenting them in court as evidence. In addition, presenting the results in a graphical format could have provided a different perspective in assembling the conclusion, which could have improved the results.

7.10 User Experience Testing and Analysis

To optimise the result on the system, a systematic and regulated approach to testing was conducted by seasoned specialists and those associated in the field of computer forensics, anti-malware practices, information security and other related areas. The testing was based on the critical metrics (parameters) by using questionnaire-based prototype testing. *User experience* is the capability of the prototype to be understood by the user in a specified condition. It is also the extent to which a prototype can be tested by specified users to achieve results, effectiveness and rate of satisfaction.

The test analysis was undertaken on the basis of reading scales based on the following five subscales: *flexibility, usability, efficiency, learnability and helpfulness.* These scales are based on the criteria outlined in the measure of software quality in its development (Bevan, 1995):

- Flexibility – the degree to which the prototype can be modified on the basis of the needs and requirements of the user. It also shows the extent to which the prototype can be scalable;
- Usability – a scale to check the ease of use of the prototype;
- Efficiency – the degree to which a user can achieve his or her objective of working on the prototype in a direct and recorded time;
- Learnability – the degree of learning and adapting new things, such as features, in the prototype and providing an affirmative reaction; and
- Helpfulness – the degree to which the prototype assists the user with executing the

prototype in the most effective manner.

A total of 30 respondents were called upon to evaluate the system. Their evaluation response was graded using a scale of 1 to 5. A rating of 5 was the most positive response and a rating of 1 was the most negative response against the parameters as follows:

5 strongly agree . . . 4 . . . 3 . . . 2 . . . 1 strongly disagree.

An analysis was carried out on the responses to obtain the results. The results of the analysis are shown in Table 14:

Parameter	**Rating**				
	5	**4**	**3**	**2**	**1**
Flexibility	19	7	3	1	0
Usability	13	9	5	2	1
Efficiency	14	5	7	4	0
Learnability	26	4	0	0	0
Helpfulness	18	3	6	1	2

Table 14: Results of User Experience Questionnaire on the Toolkit

Observation and Analysis

Flexibility – The majority of the respondents were in favour of flexibility of the system. The system provides a platform for users to perform dynamic scaling on the basis of their needs and requirements, which adds to the overall scope and flexibility of the Toolkit. Depending on the

investigative environment and the malware behaviour, users can select and modify the Toolkit to satisfy the environmental attributes. Respondents were pleased with how the Toolkit could adapt to their specific requirements, and if there was a change in their requirements, see how it would still meet their needs. This demonstrates how the tool would still serve its purpose in the respondents' eyes, regardless of the application, thereby increasing its likelihood of being called upon as a tool.

Usability – The majority of the respondents felt at ease while using the software navigation panel, which demonstrates the software's user-friendliness. Therefore, it is more likely to be frequently used and by more users, if they find it something that is intuitive and easy to navigate.

Efficiency – The majority of the respondents found the process of investigation efficient when using this tool. This was because the Toolkit comprises multiple tools on a single platform, making the switching of tools straightforward and easy. This feedback demonstrates how a single port of call, in this case the software under investigation, negates the need for different tools on different platforms. As a multitude of tools are available to the user and navigating to each tool was found to be easily done, time saving would be possible. It also highlights how the amalgamation of different tools in a single platform will remove the need for multiple tools across multiple platforms and, thus, in theory would save on outlay costs.

Learnability – Of the respondents, 26 out of 30 strongly agreed that they picked up this system quickly without any trouble, indicating that it is adaptable and acceptable towards any new attribute. Quickly adapting to a new system that a person has not used before identifies it as something people are more likely to use in the future as no prior learning is required, allowing the user to simply get on with the task at hand. Systems that are complex or take time to learn will be seen as difficult and will most likely put off users in favour of easier and more intuitive

systems where time will be saved by not having to learn about them.

Helpfulness – Of the respondents, 18 out 30 felt strongly that the system provided the expected results with the expected rationale. This showed that the system provided a toolset that was required by the users to carry out the intended job, thereby giving users access to what they expected and needed. The system, therefore, demonstrates how it serves its purpose when carrying out investigative procedures and ultimately has value.

The results for the questionnaire were further analysed using percentages to better group the responses as either positive or negative factors. Any rate above 2 was considered positive while 2 or less was negative. These results are detailed in Table 15 below with a graphical representation of the same data shown in Figure 32.

Table 15: Statistical Results for User Experience Questionnaire

Factor	**Parameter**				
	Flexibility	**Usability**	**Efficiency**	**Learnability**	**Helpfulness**
Positive	97%	90%	87%	100%	90%
Negative	3%	10%	13%	0%	10%

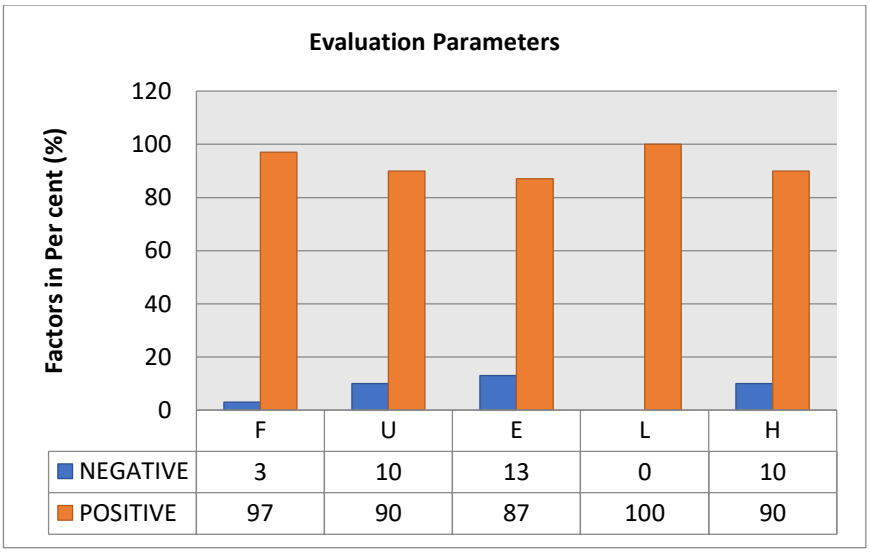


Figure 62: Statistical Results for User Experience Questionnaire

From the results in Table 15, it can be concluded that there was general positive acceptability of the prototype. For flexibility, the input user group score was a 97% positive ratio result. The other factors also scored well in the positive factor ratio, with learnability (100%), efficiency (87%), usability (90%) and helpfulness (90%) all scoring high. Conversely, all negative ratios were relatively low with the highest negative score for efficiency at 13%, while the lowest was 0% with no negative ratio for learnability. The results suggest that users learnt easily, found the experience intuitive and useful and, in general, were happy with the overall experience. The results also suggest that improvements could be made, although the amount of feedback was largely positive and did not highlight any significant negative areas.

7.11 Training Needs Analysis

A training needs analysis (TNA) is commonly undertaken by companies who recognise that training is required on a service or product, but needs to address the way in which it is applied. The release of a new software product will often require training for the end user to ensure the tool appropriately meets the users' requirements, and they are comfortable and knowledgeable

about the product.

The results outlined in the previous section showed that the user experience was well received with a high percentage of respondents determining the learnability and usability as positive. This suggests that the requirement for training is not considerable. However, it should be noted that the selected respondents were industry experts and familiar with existing tools and methodologies. It is feasible that some future users would not be experienced in their respective fields and, therefore, would be less familiar with similar software or the general processes involved in malware. Hence, a strategy should be adopted to include either instructor-based courses, web-based courses, or built-in tutorial guides to ensure the provision of adequate training.

CHAPTER 8 - CONCLUSION

8.1 Overview

Malware is a threat to all existing computer systems regardless of use and the level of security deployed on its network. As presented in the literature review (Chapter 2), malware has a global impact on the digital world, empowering different groups of individuals who pose dangers to financial services and engage in criminal activities. Thus, there is an urgent and ever-increasing need to carry out investigative processes and use specific procedures to implement preventive measures against malware. Various manual and automated techniques for malware detection and analysis are available, but they are often time-consuming and costly (see Section 2.8). When using such techniques, performance often deteriorates when investigations are carried out on large volumes of data. This is due to a lack of computer resources and inefficiencies in the detection software's ability to handle vast amounts of data. There are also various other challenges associated with investigation, detection and analysis of malware (see Section 2.11). For example, variations in scanning for malicious programs can be a problem as malware can change its behaviour heuristics, making it too complex for fixed detection methods. In some situations, investigators are unable to create matching patterns that link previously recorded malware data with new ones to obtain results of existing infected files being corrupted by other malware, thereby generating inaccurate results and affecting future detections. Moreover, in some scenarios, the cost and skills related to existing malware detection create a challenge for investigators (see Section 2.11). Thus, there was a need to develop and deploy an effective system that can assist investigators in the accurate and complete detection and analysis of malware without compromising data integrity.

To collect knowledge concerning a suitable solution for the threats and issues identified in the literature review (Chapter 2), a market analysis was conducted (Chapter 4) using industry-based

respondents to give opinions and considerations via a questionnaire and interviews. This analysis helped to identify a need for an encompassing system that can handle both the dynamic and static investigative methods. The analysis also showed that inadequacies in current software detection tools can produce inaccurate or incomplete results. Furthermore, the cost implications in the current use of multiple tools for a single investigation along with the man-hour requirements could well be reduced. Hence, the opportunity to establish a new framework to help investigators carry out effective and efficient malware detection while minimising costs was noted.

In response to this opportunity, the author demonstrated a requirement and developed a new malware framework called TCMA (Chapter 5 & Chapter 6). This newly developed TCMA framework has three tiers, namely the malware acquisition function, detection and analysis, and the database operational function. Each tier was comprehensively discussed to justify its relevance to assist in an effective and efficient malware detection and analysis process. The current challenges and associated issues with present malware detection methods were highlighted and addressed through the development and implementation of the TCMA framework.

Using the TCMA framework, the author subsequently developed a prototype Toolkit, representing a novel approach in digital forensics and malware detection methods (Chapter 6). Current tools and frameworks were shown to use either static or dynamic methods for malware detection, but the developed prototype Toolkit integrates a range of tools into a single software package; this is by using both static and dynamic analysis techniques in combination, which has not been systematically implemented before. For this Toolkit under test conditions, adopting this approach in malware has increased the accuracy rate and flexibility in the reporting structure over any individual static or dynamic method. Moreover, it readily accepts modifications to the source code using a range of programming languages, and enables reporting through various

outputs, thereby increasing usability over that of a single tool. Consideration has been given in development of the Toolkit by keeping in mind the end user, and to make the experience more intuitive and sufficiently dynamic to adapt to an individual's requirements.

As demonstrated in Chapter 7, the Toolkit underwent different phases of design, development, implementation, testing and finally assessment to demonstrate the need and result of this novel approach to malware detection. The Toolkit is simple in structure and was built using readily available open source software. It was developed to help investigators gain skills to critically respond to current and future cyber incidents and analysis. Added to this, it is capable of carrying out processes that assist in analysing infected files and checking the optimisation of available open source tools. As the Toolkit is an amalgamation and integration of various already proven open source tools, it eliminates the intention to develop a proprietary tool for malware detection. Thus, implementation focused on the development of a user-friendly solution, making the investigation process usable and scalable in any dynamic environment. Moreover, a rationale was designed on the basis of parameters to illustrate the Toolkit's superiority over a single tool for malware investigation, which showed it is a valid solution for current digital forensics. The tool was also shown to decrease costs by eliminating the requirement of different tools for investigation, reducing the number of investigative cycles, decreasing the need for extra investigators carrying out similar work and streamlining tool usage during an investigation.

The following are the primary benefits seen in the development of the TCMA framework and the Toolkit:

a) **Increased Reliability:** Due to the presence of integrity constraints in the database, updating data on malware becomes reasonably manageable. Moreover, every report is provisioned with back up, for example, putting it into a different server and encrypting the

data, thus making it highly reliable when the detection is performed through pre-existing forensic tools, pre-defined malware detection systems and limited duration at a workstation;

b) Increased Functionality: TCMA provides a platform where any new functionality can be added without losing the indispensable correlated evidence. Thus, the scope of the system can be modified with increased functionality; even in a situation where numerous systems or parties are involved in the same crime, it performs analysis independently and integrates the results to form decisions;

c) Increased Accuracy: Using both static and dynamic methods for malware detection has given an advantage to our framework. The combination of this duo has already proven to be a method for improving accuracy, rather than using static or dynamic as a singular method. Moreover, by crafting correlations among multiple cases when the investigator uses numerous techniques, it is able to form conclusions in a more effective way;

d) Reduced Costs: The proposed framework integrates highly scalable open source tools in a single package. This showed positive results in detecting and analysing large amounts of data, obfuscated malware or malicious codes, while minimising the investigative time and, consequently, costs, thereby creating scope for operational savings for the user;

e) Reduced Investigation Complexity: The observations and results (in Chapter 7 & Appendix C) showed that implementation reduces complexity in the investigation, when an illegitimate activity is carried out through the use of sophisticated tools and methods; and

f) Minimised Technique Complexity: Technique complexity occurs when a user needs to implement variable methods for malware detection or use differentiable techniques in a single investigation. The proposed framework, through using a single technique for every different environment, minimises the complications and methods associated with investigations, especially when the same data are present over a large network or in various computational systems.

The outcome of this work, the proposed artefact, is a single platform capable of multiple malware analyses that improves functionality and efficiency, while reducing costs and the time spent during an investigation. For instance, all experiments were performed with open source tools, where the results of multiple analyses were accumulated in a single user interface. Additionally, user testing showed results of better time management, as compared to previous tools used by the respondents.

8.2 Addressing Research Objectives

The goal of the research was to optimise the research paradigm for malware analysis and to improve the investigatory experience by employing an active approach to detect malware programs. With this in mind, the author divided the objectives into two phases: Phase 1 focused on developing a robust framework for malware analysis and detection, while Phase 2 involved developing an artefact as a prototype for an anti-malware Toolkit to assist investigators in systematic detection and analysis.

The objectives in Phase 1 were addressed as follows:

a) *To conduct an extensive literature review of the different trials and techniques associated with malware detection and to identify the parameters for developing the most efficient tools for a live system:*
    - A review and discussion of the literature took place, which justified the academic work undertaken by various authors in the field of malware, including methods of detection and analysis (ref: Chapter 2); and
    - The parameters were identified for developing the artefact by defining the malware detection architecture and detection techniques (ref: 2.9 The Malware Detection Architecture, 2.10 Malware Detection Techniques).

b) *To conduct extensive market research to understand the precise nature of the software/system requirements:*

- Conceptual and contextual methods were used to gather information, followed by an extensive analysis of the report (ref: Chapter 3);
- Based on the selected methodology discussed in Chapter 3, the author divided the market research into two phases: a questionnaire and an interview; and
- The questionnaire was designed with a view to developing a new effective framework for malware detection and analysis, which involved a novel approach to digital forensics and research. The interview was conducted with a view to gathering results for developing an artefact.

c) *To generate a new, effective framework to analyse the behavioural characteristics of the malware family and the mechanics of its operation:*

- A new approach (model), called the triple-tier, centralised, online, real-time environment (tri-CORE) malware analysis (TCMA) method was developed (ref: Chapter 5).

The objectives for Phase 2, developing a Toolkit, were as follows:

a) *To develop a malware detection Toolkit, representing a simple structural prototype built from open source software, to help investigators develop skills to critically respond to current cyber incidents and investigations. The development should be reflected by the information gathered from Phase 1:*

- A malware analysis Toolkit was developed, which was simple in structure yet sufficiently robust, using integrated open source malware detection software tools; and

- A rationale was designed based on certain parameters, illustrating the reasons for using the Toolkit over a single tool for malware investigation (ref: Table 9).

b) *To perform an extensive testing and recording of the Toolkit by experienced end users in order to gather real-time execution results:*

- Testing was divided into two phases: first, the experiments were performed in real time by the author (ref: 7.8 Experiments on the), and second, the end users tested the prototype (ref: 7.10 User Experience Testing and Analysis).

8.3 Research Implications

The study was conducted through distinctive cross-referencing where all possibilities were gathered, studied and analysed critically to obtain an effective result. During development of the artefact, practical implications were addressed by author. The artefact was initiated, designed and implemented by keeping the proposed framework (TCMA) in mind. It is important to note that the proposed framework is a wider contribution compared to the artefact development and, thus, the artefact was developed as an academic prototype. In the context of this academic work, the artefact was used primarily as a method to validate the framework, although it also served to demonstrate how the author believed a new product could look and behave. The artefact could have been developed into a full commercial software package, but due to limited resources, financial restrictions and time period (during an academic session), the author has not developed a finalised product. Keeping such considerations in mind, the scope was limited to developing the artefact with a limited functionality but describing all the essential attributes of the proposed framework. Thus, a complete version could be considered using the proposed TCMA framework in future work.

8.4 Future Work

The proposed prototype for malware detection and analysis, the Toolkit, is limited in its functionality. Due to theoretical and practical implications, progress can only be taken so far and further exploration and experimentation will need to be taken to finalise a product. The following are some notable features for future work:

a) A new commercial version of the artefact using the system development lifecycle (SDLC), incorporating standard design, working environment, networking infrastructure and database;
b) Conducting more experiments and investigations to detect malware in a real-time environment or on a live system to obtain more realistic results, rather than generic ones;
c) The choice and quantity of the resources could be increased in the later stage to obtain more complete results for comparison and discussion;
d) The current Toolkit is a desktop-based application; thus, a new web-based interface can be designed and developed for added flexibility;
e) The experiment was carried out in a virtual environment, which has certain implications. For instance, malware could be inactive after virtual machines are detected, so the investigator must watch for the presence of virtual machines on the hardware in order to detect a virtual machine freely running inside the system. In addition, the process is time-consuming with the need to restart the machine in any failure, although this can be improved at a later stage; and
f) The choice and number of sample malware can be increased to obtain a more comprehensive result for further analyses.

The overall result of the study is multi-faceted, whereby the objective was to discuss various effective methods for the use of malware detection and analysis. By acknowledging faults and

bottlenecks in the existing system, a new framework (TCMA) was formulated. This was developed with the intention to overcome associated limitations and assist an investigator in obtaining effective results. The study contributes to the field of IT security with the aim of making the process of malware analysis and detection more flexible and robust.

References

Andrew M (2004), Understanding open source and Free Software Licensing, O Reilly Media, Inc

Adeel,M., Tokarchuk, L.N. (2011) 'Analysis of mobile P2P malware detection framework through Cabir and Commwarrior families',in IEEE 3rd Int. Conf. Social Computing (Socialcom), 2011(pp. 1335, 1343) IEEE Publication

Adleman, L. (1990) 'An abstract theory of computer viruses', Lecture Notes in Computer Science, p.403 MIT Press

Agarwal, A., Gupta, M., Gupta, S. & Gupta, SC. (2011). Systematic digital forensic investigation model. International Journal of Computer Science and Security, Vol. 5, No. 1

Alazab, M., Layton, R., Watters, P. (2010) 'Malware detection based on structural and behavioural features of API calls', International Cyber Resilience Conference, Security Research Institute Conferences. Retrieved from http://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1000&context=icr

Alink, W., Bhoedjang, R., Boncz, P., de Vries, A. (2006) XIRAF –XML-based indexing and querying for digital forensics. Digital Investigation, vol. 3(Supplement 1), pp. 50–58.

Alshamarani A; Bahattab A (2015), A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model IJCSI International Journal of Computer Science Issues, Volume 12, Issue 1, No 1, January 2015 ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784

Aman, W. (2014). A framework for analysis and comparison of dynamic malware analysis tools. International Journal of Network Security & its Applications, Vol.6, No.5, 63-74

Amos, B., Turner, H., White, J. (2013) 'Applying machine learning classifiers to dynamic Android malware detection at scale', in Proceedings of the 9th International Wireless Communications and Mobile Computing Conference (IWCMC) (pp.1666–1671). Retrieved

from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6583806&isnumber=6583517

AV Comparatives (2013). Whole product dynamic 'real-world' protection test. [Online] Available from: http://www.av-comparatives.org/wp-content/uploads/2013/12/avc_prot_2013b_en.pdf (Accessed 7 April 2017)

AV Test (2015) [Online] Available from: https://www.av-test.org/en/statistics/malware/

Ayers, D. (2009) 'A second generation computer forensic analysis system', Journal Publication at Digital Investigation, vol. 6,pp. S34–S42.Retrieved from http://www.dfrws.org/2009/proceedings/p34-ayers.pdf

Baitsch, M., Li, N., Hartmann, D. (2010) 'A Toolkit for efficient numerical applications in Java', Advances in Engineering Software, vol. 41, no. 1, pp. 75–83.

Baker W; Goudie M (2011) Report on Data Breach Investigation. White paper document.Verizon Business.Retrieved from http://www.verizonbusiness.com/go/2011dbir

Baker W; Goudie M (2010) Report on Data Breach Investigation. White paper document. Verizon Business. Retrieved from http://www.verizonbusiness.com/go/2010databreachreport/

Barabas, M. et al. (2013) 'Automated malware detection based on novel network behavioral signatures'. Security-Oriented Research in Information &European Regional Development Fund in the IT4Innovations Centre of Excellence Project IEEE Publication

Barjtya, S; Sharma A; Rani U (2017), A detailed study of Software Development Life Cycle (SDLC) ModelsInternational Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 6 Issue 7 July 2017, Page No. 22097-22100

Bayer, U.,Kirda, E., Kruegel, C. (2010)'Improving the efficiency of dynamic malware analysis', in SAC'10 March 22–26, 2010, Sierre, Switzerland ACM'10.

Bevan, N. (1995). Measuring usability as quality of use. Software Quality Journal, Vol. 4 15-150

Bhavani, T. (2009) 'Data mining for malicious code detection and security applications', in Web Intelligence and Intelligent Agent Technologies, WI-IAT '09. IEEE/WIC/ACM International Joint Conferences, vol. 6, no. 7, pp.15–18. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5284874&isnumber=5284807

BITS (2011) 'Malware risks and mitigation report',ITS/The Financial Services Roundtable 2011.Retrieved from http://www.nist.gov/itl/upload/BITS-Malware-Report-Jun2011.pdf

Black, L. (2015) 'SmartSniff v2.17 – capture TCP/IP packets on your network adapter'.

Blount, J.J. et al (2016). 'Adaptive rule-based malware detection employing learning classifier systems: a proof of concept', in IEEE 35th Annu. Computer Software and Applications Conf. Workshops (COMPSACW), vol., no 15-18., pp. 110, 115.

Broder, E. (2012) 'Transparent detection of computer malware using virtualization'. Massachusetts Institute of Technology (MIT). Retrieved from http://web.mit.edu/broder/Public/6.uap-report.pdf

Brumley, D. et al. (2007) 'Towards automatically identifying trigger-based behavior in malware using symbolic execution and binary analysis', in Tech. Rep.CMU-CS-07-105, Carnegie Mellon University, 2007.

Burji, S., Liszka, K.J., Chan, C. (2010) 'Malware analysis using reverse engineering and data mining tools', in Proceedings of the International Conference on System Science and Engineering (ICSSE), 1–3 July 2010 (pp.619–624). Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5551719&isnumber=5551700

Buster (2013) 'Introduction to Buster sandbox'. Retrieved from http://bsa.isoftware.nl/

Carrier, B. (2005) 'File system forensics analysis'. Addison Wesley Professional. Retrieved from http://sergiob.org/unam/DGSCA/forense/FileSystemAnalysis.pdf

Casey, E. (2011) Foundations on Digital Forensics. Ch. 1. Retrieved from: http://booksite.elsevier.com/samplechapters/9780123742681/Chapter_1.pdf

Cavallaro, L.et al. (2008)'On the limits of information flow techniques formalware analysis and containment'. Computer Science Department University of California at Berkeley. Retrieved from http://www.comp.nus.edu.sg/~prateeks/papers/saxena-dimva08.pdf

Čeleda, P. et al. (2010) 'Malware detection from the network perspective using NetFlow data', in 3rd NMRG Workshop NetFlow/IPFIX Usage in Network Management, Maastricht, The Netherlands, 2010 IEEE Publication

Chen, L. et al. (2012) 'A layered malware detection model using VMM', in IEEE 11th Int. Conf. Trust, Security and Privacy Computing and Communications (TrustCom), 2012 (pp. 1259, 1264).

Chen, Z., Gao, L., Kwiat, K. (2003) 'Modeling the spread of active worms', in Proceedings of IEEE INFOCOM San Francisco, CA 2003.

Chen, Z.,Chen, C., Ji, C. (2007) 'Understanding localized-scanning worms', in Proceedings of IPCCC'07 (IEEE-International Performance Computing and Communications Conference), New Orleans, LA.

Cheok, R. (2014) 'Wireshark: A guide to Color My Packets – detecting network reconnaissance to host exploitation'. SANS Institute InfoSec Reading Room. Retrieved from https://www.sans.org/reading-room/whitepapers/detection/wireshark-guide-color-packets-35272

Christie, C.J. (2006) 'Former UBS computer system manager gets 97 months for unleashing 'logic bombs' on company network. US DOJ Press Release (News).

Church, R.M. (2001) 'The effective use of secondary data', Learning and Motivation, vol. 33, pp. 32-45.doi:10.1006/lmot.2001.1098

Clarke, N,. et al. (2012) Proceedings of the seventh international workshop on digital forensics & incident analysis WDFIA 2012,

Cobb, M. (2013) 'Keys to a successful network-based malware detection deployment'. TechTarget Powered by IT Central Station. Retrieved from http://searchsecurity.techtarget.com/tip/Keys-to-a-successful-network-based-malware-detection-deployment

Cooke, E., Jahanian, F., McPherson, D. (2005) 'The zombie round up: understanding, detecting and disrupting botnets', in Proceedings of Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI '05), Cambridge, MA, July 2005.

Coviello, A.W. (2011) RSA Written Testimony. US House of Representatives, the Security Division of EMC, CA USA

Crabtree, B.F., Miller, W. L. (eds.) (1992). Doing Qualitative Research. Newbury Park, CA: Sage.

Curtis, K.R. (2008) Conducting Market Research Using Primary Data. Whitepaper by the Department of Resource Economics, University of Nevada.

Damodaran, A. (2015). Combining Dynamic and Static Analysis for Malware Detection, Unpublished Master's dissertation, San Jose State University

Daryabar, F.et al. (2011) 'Investigation of malware defence and detection techniques', Int. J. Digital Inf. and Wireless Communication., vol. 1, no. 3, pp. 645–650.

Davis, T. (2009) 'Utilizing entropy to identify undetected malware'. Guidance Software, Cyber Security Solution. Retrieved from http://image.lifeservant.com/siteuploadfiles/VSYM/99B5C5E7-8B46-4D14-A53EB8FD1CEEB2BC/43C34073-C29A-8FCE-4B653DBE35B934F7.pdf

Demme,J, Maycock M; Tang A (2013) 'On the feasibility of online malware detection with performance counters'. Department of Computer Science, Columbia University, NY. Retrieved from http://www.cs.columbia.edu/~jdd/papers/isca13_malware.pdf

Dickens, P., Thakur, R. (2000) An Evaluation of Java's I/O Capabilities for High-Performance Computing. San Francisco, CA: Java.

Dilkina, B.,Gomes, C.P., Sabharwal, A. (2009) 'Backdoors in the context of learning', inProceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing (SAT 2009)(pp. 73–79). Swansea, UK.

Distler, D. (2007) 'Malware analysis: an introduction'. SANS Institute InfoSec Reading Room. Retrieved from https://www.sans.org/reading-room/whitepapers/malicious/malware-analysis-introduction-2103

Dittrich, D. & Bailey, M. (2013) Applying ethical principles to information and communication technology research: A companion to the Menlo Report. US Dept of Homeland Security.

Dizon, J; Galang, L; Cruz , M (2010) "Understanding WMI Malware", Whitepaper by TRENT Micro Research Paper, URL: http://la.trendmicro.com/media/misc/understanding-wmi-malware-research-paper-en.pdf

Dowling, A (2016). 'The Sleuth Kit v2.01 and autopsy forensic browser demonstration'. White paper on TSK Manual. Retrieved from http://www.sjones.co.nz/downloads/Files/Forensics/TSK_v201_Demonstration.pdf

Driscoll, D.L. (2011) 'Introduction to primary research: observations, surveys, and interviews', in Writing Spaces: Readings on Writing, Volume 2. Retrieved fromhttp://www.parlorpress.com/pdf/driscoll--introduction-to-primary-research.pdf

DTCC (2014) A White Paper to the Industry on Systematic Risk: Cyber Risk – A Global Systematic Threat. Retrieved from http://www.dtcc.com/~/media/Files/Downloads/issues/risk/cyber-risk.pdf

Duhan, N., Sharma, A.K., Bhatia, K.K. (2009) 'Page ranking algorithms: a survey', in Advance Computing Conference, IACC 2009. IEEE International(pp.1530–1537). Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4809246&isnumber=4808969

Edem, E.I., Benzaid, C., Al-Nemrat, A., Watters, P. (2014) 'Analysis of malware behaviour: using data mining clustering techniques to support forensics investigation', in Proceedings of the Fifth Cybercrime and Trustworthy Computing Conference (CTC), 24–25 November 2014(pp.54–63) IEEE Publication.

Egele, M., Antipolis, S., Kruegel, C. (2010) 'A survey on automated dynamic malware analysis techniques and tools. Paper by International Secure System Labs. Retrieved from https://iseclab.org/papers/malware_survey.pdf

Egele, Manuel, et al (2015). "On the Security and Engineering Implications of Finer-Grained Access Controls for Android Developers and Users." Detection of Intrusions and Malware, and Vulnerability Assessment: 12th International Conference, DIMVA 2015, Milan, Italy, July 9-10, 2015, Proceedings. Vol. 9148. Springer, 2015.

Falliere, N., Murchu,O.L., Chien, E. (2011) W32.stuxnet Dossier (Version 1.4). Technical report. Retrieved from http://www.symantec.com/content/en/us/enterprise/media/security_respons%e/whitepapers/w32_stuxnet_dossier.pdf, February 2011

Farmer, D. Venema, W. (2004). Forensic Discovery. Addison Wesley Professional, USA

Fender, J., Young, C. (2013) 'A practical guide to the front end Web development', Front-End Fundamentals.

Florêncio, D., Herley, C. (2011) Where Do All the Attacks Go? Economics of Information Security and Privacy III New York: Springer.

Fortinet 'Understanding how file size affects malware detection'. Fortinet. Retrieved from http://www.fortinet.com/sites/default/files/whitepapers/MalwareFileSize.pdf

Garfinkel, S. (2007)'Anti-forensics: techniques, detection and counter measures', White paper by Naval Postgraduate School, Monterey, CA. Retrieved from http://simson.net/clips/academic/2007.ICIW.AntiForensics.pdf

Ghahrai (2008), Interative Model [Online], https://www.testingexcellence.com/iterative-model/

Gibson, D. (2015). Effective Help Desk Specialist Skills. Indianapolis, IN: Pearson Education

Gross, Brett, (2009). "Your botnet is my botnet: analysis of a botnet takeover." Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009.

Gupta, B. et al. (2016). Handbook of research on modern cryptographic solutions for computer and cyber security. IGI Global

Han, S., Keungi, L., Sangjin, L. (2009) 'Packed PE file detection for malware forensics', in Computer Science and its Applications, 2nd International Conference, vol. 1, pp. 1–12. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5404211&isnumber=5404169

Hunt, R., Slay, J. (2010). 'Achieving critical infrastructure protection through the interaction of computer security and network forensics', in Privacy Security and Trust (PST), Eighth Annual International Conference, vol.23, 17-19.

IBM (2013) The Thriving Malware Industry: Cybercrime Made Easy. IBM Software Thought Leadership white paper. Retrieved from http://www.draware.dk/fileadmin/IBM/The_thriving_malware_industriy._Cybercrime_made_easy.pdf

iFrame HTML (2016), iFrame Security [online], http://www.iframehtml.com/iframe-security.html

Iliopoulos, D, et al (2011). Darwin inside the machines: malware evolution and the consequences for computer security.

Intel Security (2014) Net losses; estimating the global cost of cybercrime, centre for strategic and international studies, Intel security, 2014

Intel Security (2016). McAfee Labs Threats Report. [Online] Available from: https://www.mcafee.com/au/resources/reports/rp-quarterly-threats-dec-2016.pdf (Accessed 03/04/17)

Internet Live Stats [Online] Available from: http://www.internetlivestats.com/ (Accessed 03/04/17)

Jakobsson, M. (2008) Iframe Attacks, An Examinationof the Business of iFrame Exploitation. The Verisign iDefence Malicious Code Operations Team (2008)

Jakobsson, M., Ramzan, Z. (eds.) (2008) Crimeware: Understanding New Attacks and Defenses. Safari Technical Books.

Jamil, D. & Khan, MNA. (2011). Is ethical hacking ethical? International Journal of Engineering Science and Technology Vol.3, No. 5

John, L.J. (2012) 'Digital forensics and preservations', DPC Technology Watch Report 12-03 November, Digital Preservation Coalition 2012.

Justin, C., Klein, K. (2011) 'Anatomy of a modern malware attack', White paper proceedings at University of Pennsylvania Schools of Arts and Science (p. 10).

Kaspersky (2014) Mobile cyber threats, Kaspersky lab & Interpol joint report, 2014

Kaspersky (2017) Kaspersky security network, Big data-powered security report, 2017

Kent, K., Chevalier, S., Grance, T., Dang, H. (2006) NIST Document on Guide to Integrating Forensic Techniques into Incident Response, Special Publication 800-86. Retrieved from: http://csrc.nist.gov/publications/nistpubs/800-86/SP800-86.pdf

Keragala, D., (2016) 'Detecting malware and sandbox evasion techniques' SANS institute InfoSec Reading Room, 2016

Kolbitsch,C. et al. (2010) 'Effective and efficient malware detection at the end host'. Secure Systems Lab, TU Vienna, Institute Eurecom, Sophia Antipolis. Retrieved from https://www.usenix.org/legacy/event/sec09/tech/full_papers/kolbitsch.pdf

Kolosnjaji, B., Zarras, A., Lengyel, T., Webster, G. & Eckert, C. (2016). Adaptive Semantics-Aware Malware Classification. In J. Caballero, U. Zurutuza & R. Rodriguez (eds) , Detection of Intrusions and Malware, and Vulnerability Assessment. San Sebastian, Spain: Proceedings of 13th International Conference, 7-8 July 2016

Kozlowski, A. (2014) ‘Comparative analysis of cyberattacks on Estonia, Georgia and Kyrgyzstan’, European Scientific Journal, vol.3.

Kramer, S., Bradfield, J.C. (2008) ‘A general definition of malware’. Paper presented at the Workshop on the Theory of Computer Viruses (2008).

Krishan, K., Himanshu, U., Ritesh, K. (2012) ‘Trojan horse: infection and precaution’, BPR Technologia: A Journal of Science, Technology & Management, vol.1.

Kroll (2016) *Global Fraud and Risk report,* Annual Edition 2016-17URL: http://www.benevia.nl/wp-content/uploads/2017/03/RPT_KRL_US_Kroll_Global_Fraud_and_Risk_Report_2016.pdf

Kulshrestha, S., Sachdeva, S. (2014) ‘Performance comparison for data storage:Db4o and MySQL databases’. Department of Computer Science and Engineering, Jaypee Institute of Information Technology, Noida, India.

Kurose, J., Ross, K. (2009) ‘Wireshark lab: getting started’, Computer Networking: A Top Down Approach, 5th ed.

Lavesson, N., Boldt, M., Davidsson, P., Jacobsson, A. (2011) ‘Learning to detect spyware using end user license agreements’, Knowledge and Information Systems, vol. 26, no. 2, pp. 285–307.

Lee, J. (2013).‘Oracle vs. MySQL vs. SQL Server: a comparison of popular RDBMS.

Li, Z. et al. (2008) ‘Automatic generation of infection signatures’, in Conf. Dependable Systems and Networks (DSN), IEEE Publication 2008.

Lindorfer, M., Kolbitsch, C., Comparetti, P. (2011) ‘Detecting environment-sensitive malware’, Secure Systems Lab, Vienna University of Technology. Retrieved fromhttp://www.syssec-project.eu/m/page-media/3/disarm-raid11.pdf

Liu, W. et al. (2011) ‘Behavior-based malware analysis and detection,’ in Int. Workshop Complexity and Data Mining, 2011(pp. 39–42) IEEE Publication.

Ma, W. et al. (2010) ‘Shadow attacks: automatically evading system-call-behavior based

malware detection'. Department of Computer Science and Engineering, Texas A&M University College Station, TX. Retrieved from http://faculty.cs.tamu.edu/guofei/paper/ShadowAttacks_final-onecolumn.pdf

Maheshwari, N. (2010). Botnets – secret puppetry with computers. Academic Press.

Malwarebytes (2017). State of Malware Report. [Online] Available from: https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf (Accessed 7 April 2017)

Manar, A,. (2016). Towards early software reliability prediction for computer forensic tools (case study)

Marshall, M.N. (1996) Sampling for Qualitative Research. Oxford: Oxford University Press.

Matrosov, A., Rodionov, E., Harley, D., Malcho, J. (2011) Stuxnet Under the Microscope (Revision 1.31). Technical report. Retrieved from www.go.eset.com/us/resources/_white-papers/Stuxnet_Under_the_Microscope.pdf

McAfee (2013) The economic impact of cybercrime and cyber espionage, center for strategic and international studies

McAfee (2017) threats predictions., McAfee labs report

McRee, R. (2012) 'Buster sandbox analyzer'. White paper by ISSA Journal. Retrieved from https://holisticinfosec.org/toolsmith/pdf/may2012.pdf

Mell, P., Kent, K., Nusbaum, J. (2005) 'Guide to malware incident prevention and handling'. Recommendations of the National Institute of Standards and Technology, NIST Special Publications. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-83/SP800-83.pdf

Mickolus, E,. (2015). The counter intelligence chronology, spying by and against the united states from the 1700s through 2014. McFarland & Company Inc.

Microsoft.net (2017). Developing and deploying web services in .NET. [Online] Available from: http://inf-server.inf.uth.gr/mavcourses/wis/wp-content/uploads/sites/11/2015/05/Net-Web-Services.pdf (Accessed 6 April 2017)

Milenkovic, A., Jovanov,E. (2005) 'Using instruction block signatures to counter code injection attacks', ACM SIGARCH Computer Architecture News, vol. 33, pp. 108–117.

Moser, A. et al. (2007) 'Limits of static analysis for malware detection', in 23rd Annu. Computer Security Applications Conf., vol: 10-14 no., pp. 21, 430

Moser, A. Kruegel C (2007) 'Exploring multiple execution paths for malware analysis', Proc. IEEE Symp. Security and Privacy, IEEE Publication 2007.

Moser, A., Kruegel, C., Kirda, E. (2007) 'Limits of static analysis for malware detection. Secure Systems Lab Technical University Vienna. Retrieved from https://iseclab.org/papers/staticanalysis.pdf

Mukamurenzi, M.N. (2008) Storm Worm: A P2P Botnet, NTNU (Norwegian University of Science and Technology) white paper.

Nadiammai, G,. Hemalatha, M,. Effective approach toward Intrusion Detection System using data mining techniques, Egyptian Informatics Journal, Volume 15, Issue 1, 2014, p 37-50

Nasa (2015). Checking file integrity. [Online] Available from: https://www.nas.nasa.gov/hecc/support/kb/checking-file-integrity_243.html (Accessed 7 April 2017)

Neelakantanand,S., Rao, M. (2008) 'Threat-aware signature based intrusion-detection approach for obtaining network-specific useful alarms', in Internet Monitoring and Protection: The 3rdIntl.Conf., vol. 2, no. 3, pp. 80–85.

Newsome, J. et al. (2005) 'Polygraph: automatically generating signatures for polymorphic worms', in IEEE Symp. Security and Privacy, IEEE Publication 2005.

Nguyen, H., Franke, K., Petrovic, S. (2010) 'Improving effectiveness of intrusion detection by correlation feature selection', in Proceedings of the 2010 International Conference on Availability, Reliability and Security (ARES) (pp. 17–24) IEEE Pages-17-24. 2010.

NirSoft (2015).'SmartSniff v2.25 – capture TCP/IP packets on your network adapter'. Retrieved from http://www.nirsoft.net/utils/smsniff.html

Obasanjo, D. (2013). A Comparison of Microsoft C# Programming Language to Sun Microsystems Java Programming Language. Whitepaper.

Oprisa, C., Checiches, M., Nandrean, A. (2014) 'Locality-sensitive hashing optimizations for fast malware clustering', in Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference, vol., no., pp.97–104, 4-6 Sept. 2014 IEEE Publication.

Park, S. (2012) 'Malware expert: execution tracking', in 3rd Cybercrime and Trustworthy Computing Workshop (CTC), vol. 48, no. 55, pp. 29–30 IEEE Publication.

Pteresen, K; Wohlin C; Bac D (2009), The Waterfall Model in Large Scale Development, International Conference on Product-Focused Software Process Improvement

Perdisci, R., Lanzi, A., Lee, W. (2008) 'McBoost: boosting scalability in malware collection and analysis using statistical classification of executables', in Proceedings of the Annual Computer Security Application Conference (ACSAC)(pp. 301–310).Anaheim, CA: ACM Press.

Pollard, J. & Mak, K. (2016). The Forrester Wave: Automated Malware Analysis, Q2 2016. Cambridge, MA: Forrester Research

Poulsen, Kevin (2013), Slammer worm crashed Ohio nuke plant network, Security Focus, available online at: http://www.securityfocus.com/news/6767

Provos, N., McNamee, D, Mavrommatis, P., Wang, K. & Modadugu, N. (2007). The ghost in the browser analysis of web-based malware. Cambridge, MA: Proceedings of 1st Conference on Hot Topics in Understanding Botnets

Qing, H., Dinev, T. (2005) ‘Is spyware an Internet nuisance or public menace?’, Communications of the ACM, vol. 48, no. 8, pp. 61–66.

Rahmatian, M., Kooti, H., Harris, I.G., Bozorgzadeh, E. (2012) ‘Hardware-assisted detection of malicious software in embedded systems’, Embedded Systems Letters, IEEE, vol. 4, no. 4, pp.94–97. Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6301679&isnumber=6377156

Rastogi, V. (2013). Evaluating android anti-malware against transformation attacks. [Online] Available from: http://pages.cs.wisc.edu/~vrastogi/static/papers/rcj13a.pdf (Accessed 7 April 2017)

Reiter, M., Yen, T. (2008) ‘Traffic aggregation for malware detection’, in Conf. Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA), 2008.

Rekhis, S., Boudriga, N. (2011) ‘Logic-based approach for digital forensic investigation in communication networks’, Computers & Security, vol. 30, pp. 376–396.

Richard, G.G. III, Roussev, V. (2006) ‘Next-generation digital forensics’, Community ACM, vol. 49, no. 2,pp. 76–80.

Richter, C. (2016). The Shifting Botnet Landscape: Threats and Discovery Techniques. RSA Conference, Singapore 20 – 22 July

Rieck, K,. (2011). Automatic analysis of malware behavior using machine learning

Rieck, K., Trinius, P., Willems, C., Holz, T. (2011) ‘Automatic analysis of malware behavior using machine learning’, Journal of Computer Security. Retrieved fromhttp://www.covert.io/researchpapers/security/Automatic%20Analysis%20of%20Malware%20Behavior%20using%20Machine%20Learning.pdf

Rieck, K., Willems, T., Düussel, P., Laskov (2008) ‘Learning and classification of malware behaviour’, in 5th International Conference on Detection of Intrusions and Malware, and

Vulnerability Assessment (pp. 108–125). Berlin: Springer-Verlag. http://niels.xtdnet.nl/papers/practical.pdf

Rieck, K., Willems, T., Düussel, P (2008) 'Learning and classification of malware behaviour', in 5th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment(pp. 108–125). Berlin: Springer-Verlag.

Robert Simmons (2016), Open Source Malware Lab, Virus Bulletin Conference October 2016, https://www.virusbulletin.com/uploads/pdf/magazine/2016/VB2016-Simmons.pdf

Ruparelia (2010), Software Development Lifecycle Models, ACM SIGSOFT Software Engineering Notes, May 2010 Volume 35 Number 3,

SANS (2007) [Online] Available from: https://www.sans.edu/cyber-research/security-laboratory/article/log-bmb-trp-door

SANS Institute (2009), Building an Automated Behavioral Malware Analysis Environment using Open Source Software, https://www.sans.org/reading-room/whitepapers/tools/building-automated-behavioral-malware-analysis-environment-open-source-software-33129

Schipka, M (2007), The Online Shadow Economy: A Billion Dollar Market For Malware

Authors, *Message labs, White Paper.*

URL:https://www.legis.iowa.gov/DOCS/LSA/IntComHand/2009/IHEGC012.PDF

Schultz, E. (2003) 'Pandora's box: spyware, adware, autoexecution, and NGSCB', Computers & Security, vol. 22, no. 5, pp. 366–367.

Schweitzer, D. (2002) Securing the Network from Malicious Code: A Complete Guide to Defending Against Viruses, Worms, and Trojans.Wiley Publishing, Inc. NY USA

Shannon, C.E. (1948) ‘A mathematical theory of communication’, Bell System Technical Journal 27, p.379-423.

Sharif, M. et al. (2008) ‘Impeding malware analysis using conditional code obfuscation’, in Proc. Annu. Network and Distributed System Security Symp. NDDS, 2008.

Sheriff, P. et al (2017) The business value of .NET. [Online]. Available from: https://msdn.microsoft.com/en-us/library/ms953309.aspx (Accessed 6 April 2017)

Shijo, PV; Salim, A (2015), Integrated Static and Dynamic Analysis for Malware Detection, *in proceeding Computer Science 46:804-811,* from https://www.researchgate.net/publication/276109044_Integrated_Static_and_Dynamic_Analysis_for_Malware_Detection

Siddiqui, M., Wang, M.C., Lee, J. (2008) ‘A survey of data mining techniques for malware detection using file features’, in Proceedings of the 46th Annual Southeast Regional Conference, New York, NY (pp. 509–510) IEEE Publication.

Sijtsma, K., Verweij, A. (1992) ‘Mokken scale analysis: theoretical consideration and an application to transitivity tasks’, Applied Management in Education, vol. 5, no. 4, pp. 355–373.

Smith, M. (2010) ‘Monitor your computer better with Process Hacker’. Retrieved from http://www.makeuseof.com/tag/monitor-computer-process-hacker/

Source Forge (2015) ‘Overview on Process Hacker’. Retrieved from http://processhacker.sourceforge.net/

Sponchioni, R. (2010) ‘RMAS (run-time malware analysis system):a framework for malware analysis and malware detection’. Whitepaper by Kaspersky Labs. Retrieved from http://www.kaspersky.com/images/sponchioni,_roberto_-_rmas_a_framework_for_malware_analysis_and_malware_detection.pdf

Stephen, M.., Lee, R.B. (2004) ‘Distributed denial of service: taxonomies of attacks, tools, and countermeasures’, in Proceedings of the 17th International Conference on Parallel and

Distributed Computing Systems, 2004 International Workshop on Security in Parallel and Distributed Systems(pp. 543–550).

Symantec (2016) Internet security threat report, volume 21,

Szor, P. (2005). The Art of Computer Virus Research and Defense. Addison Wesley Professional Pearson Education.

Taboada, G.L., Tourino, J., Doallo, R. (2009) 'Java for high performance computing: assessment of current research and Practice', in Proc. 7th International Conference on the Principles and Practice of Programming in Java (PPPJ'09), Calgary, Alberta, Canada, 2009 (pp. 30–39).

Thuraisingham, B. (2011) 'Data mining for malicious code detection and security applications', in Proceedings of the Intelligence and Security Informatics Conference (EISIC)(pp. 4–5 & 12–14). Retrieved from http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6061180&isnumber=6061109

Tratt, L. (2011) 'What role for static analysis in malware detection?', Slides from Middlesex University. Retrieved from http://crest.cs.ucl.ac.uk/cow/12/slides/LaurieTratt_cow_apr_2011.pdf

Trinius, P. et al. (2009) 'A malware instruction set for behaviour-based analysis', Technical report, University of Mannheim.

Tsai, M.-J., Wang, C.-S., Liu, J., Yin, J.-S. (2012) 'Using decision fusion of feature selection in digital forensics for camera source model identification', Computer Standards & Interfaces, vol. 34, pp. 292–304.

Turner, D.W. (2010) 'Qualitative interview design: a practical guide for novice investigators', Qualitative Report, vol. 15.

Tzermias, Z,, Sykiotakis, G., Polychronakis, M. & Markatos, E. (2011). Combining Static and Dynamic Analysis for the Detection of Malicious Documents, In proceedings of the 4th European Workshop on System Security, April 2011

TzeTzuen, Y.,Dehghantanha, A., Seddon, A. (2012) 'Greening digital forensics: opportunities and challenges', Signal Processing and Information Technology, vol.1, pp. 114-119.

Vemparala, S. (2015) 'Malware detection using dynamic analysis'. Faculty of the Department of Computer Science San Jose State University. Retrieved fromhttp://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1403&context=etd_projects

Vermesan, O. & Friess, P. (2014). Internet of Things Applications: From Research and Innovation to Market Deployment. Gistrup, Denmark: River Publishers

Vinod, P., Laxmi, V (2009) Survey on malware detection methods'. Department of Computer Engineering, Malaviya National Institute of Technology. Retrieved from http://www.security.iitk.ac.in/contents/events/workshops/iitkhack09/papers/vinod.pdf

Volynkin, A. VA Skormin. (2006) 'Evaluation of run-time detection of self-replication in binary executable malware', in IEEE Information Assurance Workshop (pp. 184–191).

Vuong, Son T., and Mohammed S. Alam (2011) Advanced Methods for Botnet Intrusion Detection Systems. INTECH Open Access Publisher, 2011.

W3C-ONLINE. XML query (xquery) - http://www.w3.org/xml/query/

Wangen, G (2015), The role of malware in reported cyber espionage: a review of the impat and mechanism,. Norwegian information security laboratory, Centre for cyber and information security, Gjovik University College, Teknologivn.

Waterfall Model (2012) (Online) International Journal of Engineering & Technology (iJET), ISSN: 2049-3444, Vol. 2, No. 5, 2012

Websense (2014) White Paper on Advanced Persistent Threats and the Advanced Attacks, Threats Analysis and Defense Strategies for SMB, Mid-Size and Enterprise Organizations. Retrieved from https://www.websense.com/assets/white-papers/whitepaper-websense-advanced-persistent-threats-and-other-advanced-attacks-en.pdf

Whittaker, C, (2016) Malware workshop presentation, APACS, APEC-OECD,

Wilson, D (2017) White Paper (Online) https://www.law360.com/internationaltrade/articles/553621/solarworld-urges-chinese-hacking-probe-in-trade-row

Wolff, Chisholm (2015) 'Structural entropy analysis for automated malware classification', in RSA Conference 2015, San Francisco.

Xu, J; AH Sung (2008) 'Polymorphic malicious executable scanner by API sequence analysis', in 4th Intl. Conf. HIS (pp. 378–383).

Ye, Y., Li, T., Jiang, Q., Wang, Y. (2010) 'CIMDS: adapting post processing techniques of associative classification for malware detection', IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol.40, no.3, pp.298–307.

Yin, H (2007) Panorama: Capturing system-wide information flow for malware detection and analysis', in Proc. CCS'07.Alexandria, VA: ACM Press.

Yin, H.,Song, D.,Egele, M.,Krugel, C., Kirda, E.(2007) 'Panorama: capturing system – wide information flow for malware detection and analysis', inProc CCS'07 conference on computer science, October 29 –November 2, 2007, Alexandria, Virginia, USA.ACM Press.

Zeltser, L. (2010) Introduction to Malware Analysis. SANS Institute white paper.

Zeltser, L. (2015). 5 steps to building a malware analysis toolkit using free tools. [Online]. Available from: https://zeltser.com/build-malware-analysis-toolkit/ (Accessed 6 April 2017)

Appendices

Appendix A- Questionnaire for a New Framework

Section A: Questions based on knowledge and opinions about malware

1) How much experience do you have with modern malware?
    i. 0–1 year
    ii. 1–3 years
    iii. 3–5 years
    iv. 5–8 years
    v. More than 8 years
2) How often have you seen changes in the malware landscape?
    i. Never
    ii. Seldom
    iii. Occasionally
    iv. Frequently
3) Is your own organisation vulnerable to malware attacks?
    i. No
    ii. Yes
    iii. Not sure
4) What is the situation with other non-technical employees? Are they capable of identifying a malware attack?
    i. No
    ii. Yes
    iii. Not sure

Section B: Questions based on various security processes, adopted technologies and gained skills in malware detection

1) Do you have any expertise in the malware handling process?
    i. Yes
    ii. No
    iii. Don’t know
2) How often do you receive case studies related to malware for investigation?
    i. < 5%
    ii. 5%–20%
    iii. 20%–50%
    iv. 50%–80%
    v. Above 80%
3) How many of the above cases were solved with accurate results?
    i. < 5%
    ii. 5%–20%
    iii. 20%–50%
    iv. 50%–80%
    v. Above 80%
4) What is your opinion on the level of skills and technological tools adopted in your company to carry out an investigation process on malware?
    i. Weak
    ii. Average
    iii. Strong
5) What is the level of effectiveness in detection and response?
    i. Weak
    ii. Average
    iii. Strong

Section C: Questions based on malware detection tools and analysis

1) In your opinion, what types of tools have been adopted in malware detection and analysis?
    i. Open source tools
    ii. Shareware
    iii. Commercial tools
    iv. Cannot say
2) Do you think there is a need for a new customised tool for malware detection and analysis?
    i. Yes
    ii. No
    iii. Cannot say
3) Do you think cost plays an important role in adopting an individual analysis tool?
    i. I agree
    ii. I do not agree
    iii. Cannot say
4) In your opinion, which of the following methods is more viable and effective in analysing malware?
    i. Static analysis
    ii. Dynamic analysis
    iii. Other _____
5) A) If you selected static analysis, then please specify the core adopted method for operations _____
5) B) If you selected dynamic analysis, then please specify the core adopted method for operations _____
6) Is your tool equipped with a specific preloaded database for malware?
    i. Yes
    ii. No
    iii. Not sure
7) If yes, then how frequently do you update the malware database?
    i. Every day
    ii. Weekly
    iii. Monthly
    iv. Annually
    v. Do not do updates at all

vi. Do not have any idea

8) Can you list some of the adopted tools for malware detection in your organisation?

_____

Section D: Questions based on problems faced by investigators while conducting a malware analysis

1) What is the core problem faced by the investigator during malware detection?
    i. Detecting the size
    ii. Analysing the behaviour
    iii. Detecting the level of infection
    iv. Proliferation
    v. Database updating
    vi. Other _____
2) What is the level of malware penetration in specific file processes?
    i. Critical
    ii. Complex
    iii. Complex and critical
    iv. Highly complex and critical
3) On average, how many days does it take for an investigator to detect the presence of malware in a file?
    i. One week
    ii. Less than a month
    iii. Many months
    iv. Many years
    v. Cannot say
4) What is an important parameter to look for during an investigation?
    i. A malware signature
    ii. Malware behaviour
    iii. The level of infection
    iv. All of the above

Appendix B- Interview Questions for Toolkit Market Analysis

1 Could you explain the area of your expertise in the field of cyber forensics and investigation?
2 Could you explain various categories of malware/malicious codes available in the area of information security?
3 What are the threats posed by the presence of malware on the internet?
4 Do you feel that an average user is more aware of different threats influenced by malicious code online, or is this counterpoise due to skilled malicious programmers and creators?
5 How difficult is it for investigators to detect malware, especially when the programs are newly built and affect the user activity online?
6 What sort of malware attacks are common on the internet these days?
7 Have you heard about DDoS attacks? Could you give a basic overview on that and explain the effects of DDoS on user activity?
8 Could you explain the importance of anti-malware tools?
9 What sort of anti-malware tools would you prefer as an investigator?
10 Which method of malware analysis is better – dynamic analysis or static analysis?
11 Do you find it difficult to gather information and perform analyses for malware detection?
12 What sort of emphasis should an investigator place on behavioural analysis of infected files in the system? Why is this important?
13 What are the challenges an investigator faces during malware detection?
14 Do you think we need a customised tool for detecting malware in the system?
15 What kind of changes do you recommend in the scenario of malware detection tools?
16 Do you think we need any changes in the reporting structure?
17 Do you think collecting evidence on malware is challenging these days?
18 What are the ideal attributes for developing an effective malware detection tool?

Appendix C- User Testing Program

Serial No…… **Date…….**

Name……

Note: Please read the following detailed parameters carefully and tick the appropriate option in the box on the next page. You have five options to define your experience with the system.

Flexibility – degree to which the prototype can be modified on the basis of the needs and requirements of the user. It also shows the extent to which the prototype can be scalable.

Usability – scale to check the ease of use of the prototype

Efficiency – degree to which a user can achieve his or her objective of working on the prototype in a direct and record time.

Learnability – degree of learning and adapting new things, such as features in the prototype, and providing an affirmative reaction.

Helpfulness – degree to which the prototype assists the user with executing the prototype in the most effective manner.

Please tick or cross the box ☒ on the next page

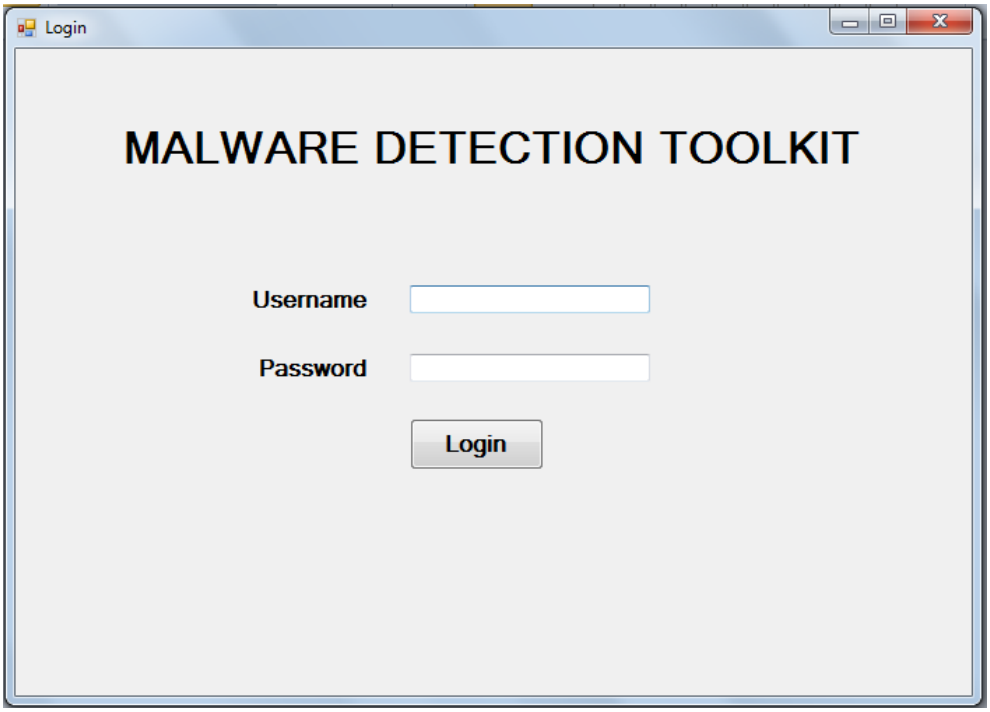
Tick 5 (if strongly agree) ...4…3…2…1 (strongly disagree)

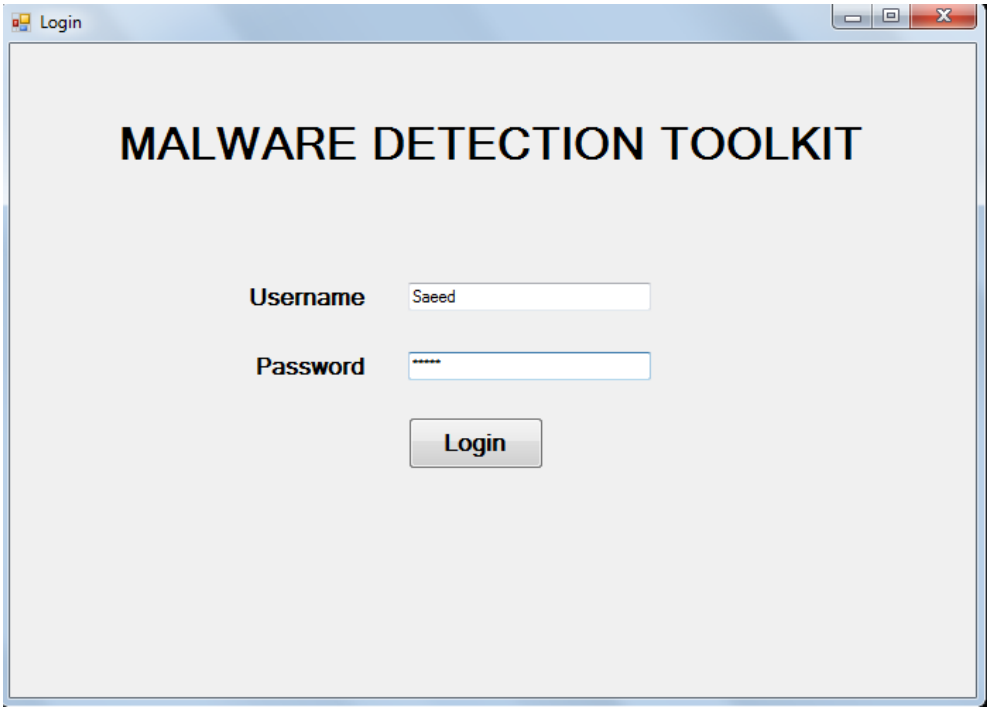
Parameter	**Rating**				
	5	**4**	**3**	**2**	**1**
Flexibility	☐	☐	☐	☐	☐
Usability	☐	☐	☐	☐	☐
Efficiency	☐	☐	☐	☐	☐
Learnability	☐	☐	☐	☐	☐
Helpfulness	☐	☐	☐	☐	☐

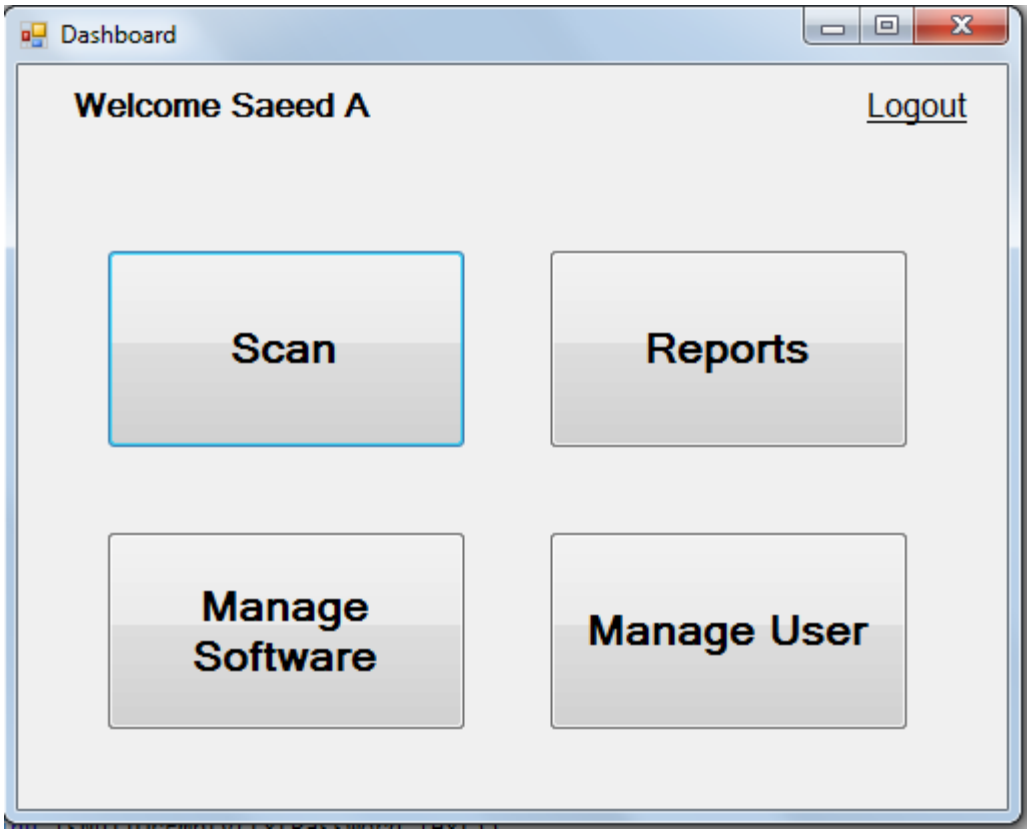
(Your feedback will only be used for academic purposes)

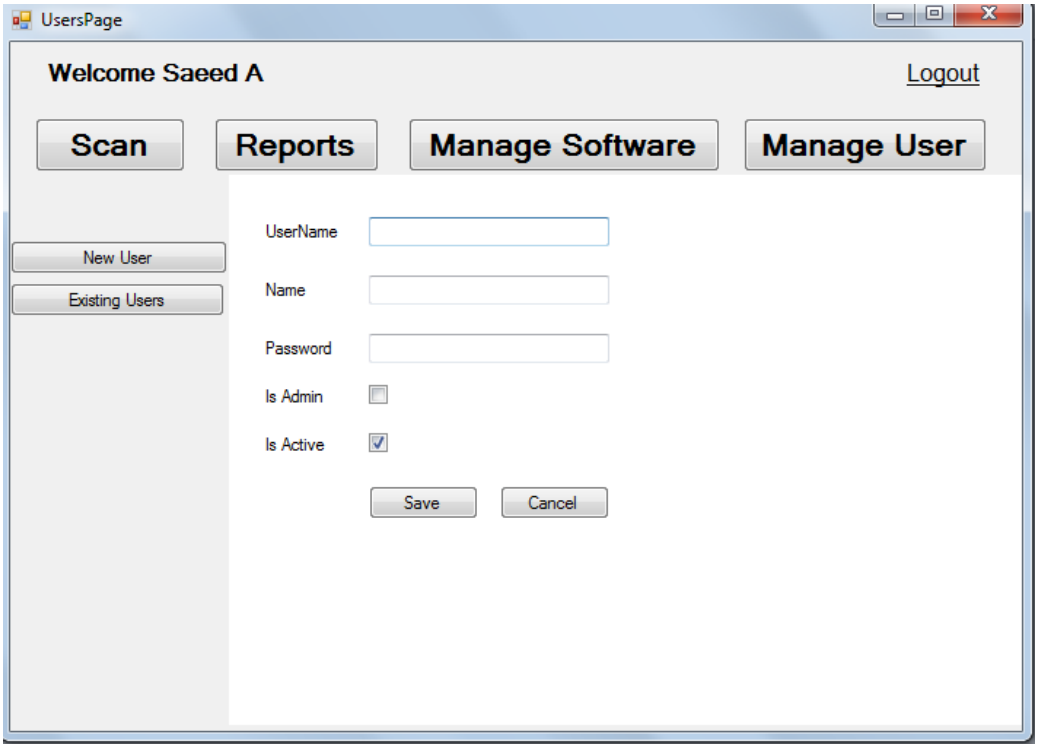
Thank you for your valuable time

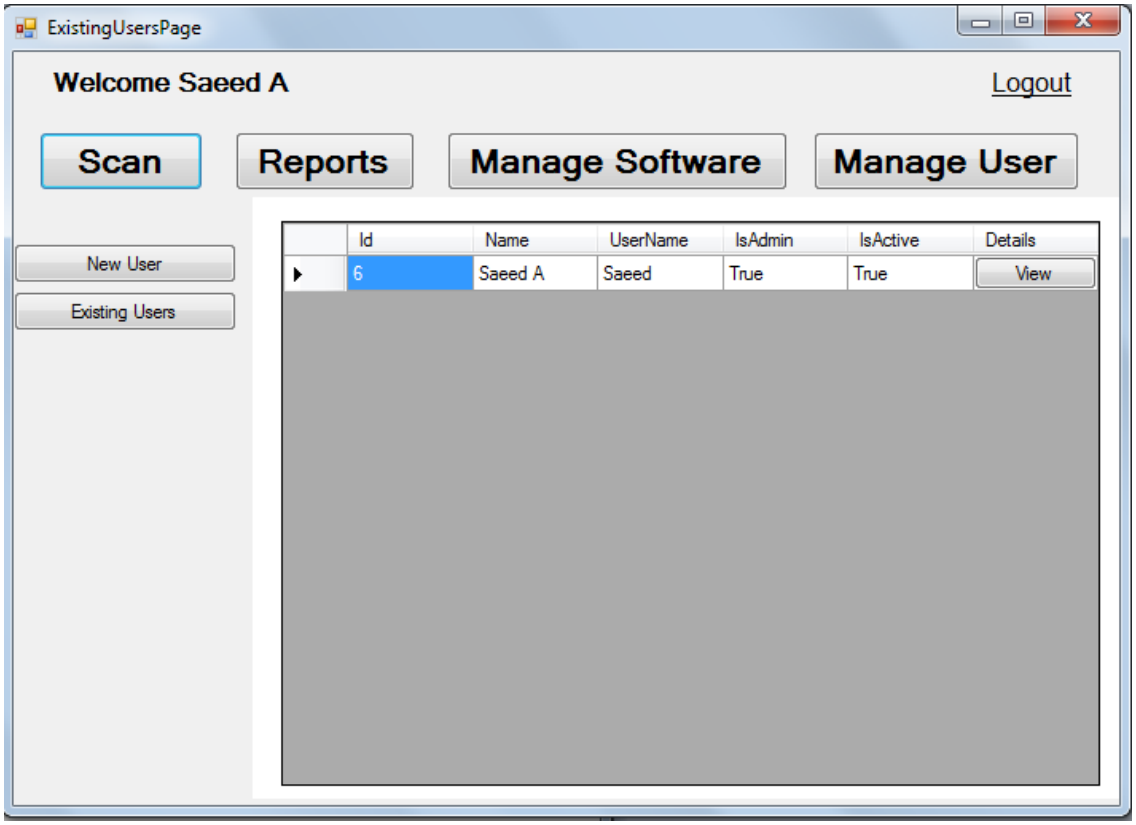
Appendix D- Screenshots of the Working Toolkit

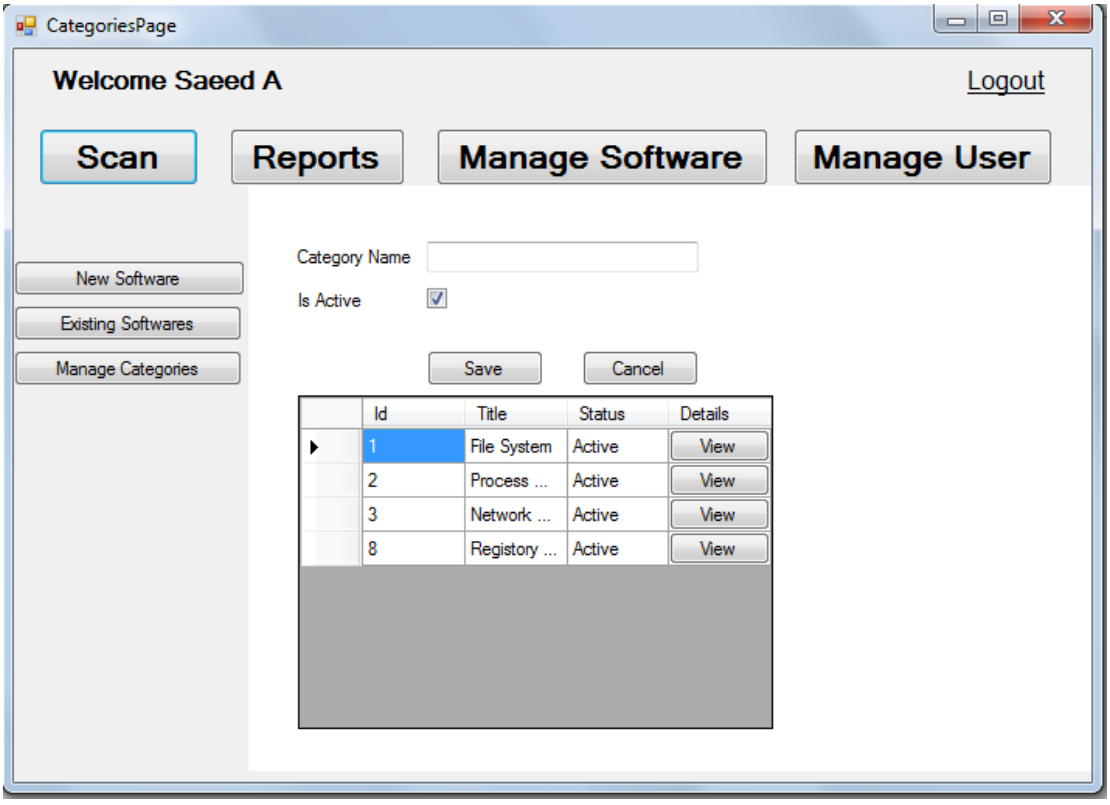





Dashboard
Welcome Saeed A
Logout
Scan
Reports
Manage Software
Manage User


UsersPage
Welcome Saeed A
Logout
Scan
Reports
Manage Software
Manage User
New User
Existing Users
UserName
Name
Password
Is Admin
Is Active
Save
Cancel


ExistingUsersPage
Welcome Saeed A
Logout
Scan
Reports
Manage Software
Manage User
New User
Existing Users
Id
Name
UserName
IsAdmin
IsActive
Details
6
Saeed A
Saeed
True
True
View


CategoriesPage
Welcome Saeed A
Logout
Scan
Reports
Manage Software
Manage User
New Software
Existing Softwares
Manage Categories
Category Name
Is Active
Save
Cancel
Id
Title
Status
Details
1
File System
Active
View
2
Process ...
Active
View
3
Network ...
Active
View
8
Registory ...
Active
View

Appendix E- Publication in the International Journal of Network Security and Its Application (IJNSA)

Source URL: http://airccse.org/journal/nsa/6114nsa01.pdf

International Journal of Network Security & Its Applications (IJNSA), Vol.6, No.1, January 2014

Optimised Malware Detection in Digital Forensics

SaeedAlmarri and Dr Paul Sant

Institute for Research in Applicable Computing, University of Bedfordshire, Luton, United Kingdom

Associate Dean, University Campus Milton Keynes, Milton keynes, United Kingdom

Abstract

On the Internet, malware is one of the most serious threats to system security. Most complex issues and problems on any systems are caused by malware and spam. Networks and systems can be accessed and compromised by malware known as botnets, which compromise other systems through a coordinated attack. Such malware uses anti-forensic techniques to avoid detection and investigation. To prevent systems from the malicious activity of this malware, a new framework is required that aims to develop an optimised technique for malware detection. Hence, this paper demonstrates new approaches to perform malware analysis in forensic investigations and discusses how such a framework may be developed.

Keywords

Denial of service (DOS), Wireshark, Nessus, TCPView, The Sleuth Kit (TSK), Autopsy, Digital Forensics, Malware analysis, Framework

1. Introduction

Over the last decade, there have been noteworthy improvements in techniques to detect malware activities [1]. Loading and distributing executable files over the Internet always presents a risk to the overall security of the system [2]. Malware programmes can be installed by attaching hidden malicious code in an innocuous file or application. The code can then be activated by a remote programmer with the aim of threatening the existing system. According to a study by Islam et al. on the risk of downloading [3], of more than 450,000 files downloaded, approximately 18% contained malware programs. They also investigated whether different code investigation techniques yielded the same results. Astonishingly, they found that there were many cases where forensic investigatory tools were unable to detect the malware content of the infected files.

A significant amount of effort has been expended on developing techniques to perform robust computer forensic investigations [6]. Such effort has focused on collecting, analysing and preserving evidence of malware activities, for e.g. a study on botnets [4] and a study of executable spyware and client-sided honeypots[5] also illustrated defensive mechanism for securing a system both on the client and server side access. Other reports mentioned in [3][6] have also focused on acquiring large and diverse samples of malware to enable researchers and forensic experts to understand their nature and its rationale. Some existing tools like ERA remover, conficker, etc. can execute hidden and anonymous files and monitor their behaviour. These tools provide protection from all threats related to the malware functioning in the system. According to reports by Kasama et al (2012), a single piece of malware can compromise and infect the entire network system. Thus, protecting systems from unwanted malicious code can be considered as one of the most critical concerns in information security [6].

DOI : 10.5121/ijnsa.2014.6101

Appendix F- Source Code for the Toolkit

User Creation

```
private void btnSave_Click(object sender, EventArgs e)
        {
            UserComponent userComponent = new UserComponent();
            if                              (userComponent.IsUserNameExists(txtUserName.Text,
StaticValues.UserEditId))
            {
                lblMessage.Text = "Username already exists.";
                return;
            }
            User usr = new User();
            usr.UserId = StaticValues.UserEditId;
            usr.UserName = txtUserName.Text;
            usr.Name = txtName.Text;
            if (StaticValues.UserEditId == 0 || !string.IsNullOrEmpty(txtPassword.Text))
                usr.Password = Security.GenerateEncryptedPassword(txtPassword.Text);
            usr.IsActive = chkIsActive.Checked;
            usr.IsAdmin = chkIsAdmin.Checked;
            usr.IsDeleted = false;
            if (userComponent.Save(usr))
            {
                System.IO.Directory.CreateDirectory(AppDomain.CurrentDomain.BaseDirectory
+ "reports\\" + txtUserName.Text);
                lblMessage.Text = "User saved successfully.";
                Clear();
            }
            else
                lblMessage.Text = "Error while saving the record.";
        }
```

Get Report

```
private void GetScanDetail()
        {
            int left = 10;
            ExecutionComponent executionComponent = new ExecutionComponent();
            ExecutionLog                                log                                =
executionComponent.GetReportById(StaticValues.ExecutionLogId);
            textBox1.Text = log.Title;
            textBox2.Text    =    !string.IsNullOrEmpty(log.Content)    ?    log.Content    :
string.Empty;
            textBox3.Text                                                                  =
executionComponent.GetKeywordsByExecutionLogId(StaticValues.ExecutionLogId);
            foreach (var VARIABLE in log.ScanScreenShots)
            {
                PictureBox picture = new PictureBox();
                picture.ImageLocation     =     AppDomain.CurrentDomain.BaseDirectory     +
"reports\\" + StaticValues.UserName + "\\" +
                                    log.Title + "\\" + VARIABLE.Image;
                picture.Size = new Size(100, 100);
                picture.Left = left;
                picture.SizeMode = PictureBoxSizeMode.StretchImage;
                picture.Click += new EventHandler(PictureClick);
                panel2.Controls.Add(picture);
                left = left + 110;
            }
        }
```

Save Report Edit Changes

```
private void button1_Click(object sender, EventArgs e)
        {
            ExecutionComponent executionComponent = new ExecutionComponent();
            if (executionComponent.Update(StaticValues.ExecutionLogId, textBox2.Text))
            {
                executionComponent.DeleteKeywords(StaticValues.ExecutionLogId);
                string[] keywords = textBox3.Text.Split(',');
```

```
                foreach (var keyword in keywords)
                {
                    ENTITIES.Keyword kw = new ENTITIES.Keyword();
                    kw.ExecutionLogId = StaticValues.ExecutionLogId;
                    kw.Name = keyword;
                    executionComponent.SaveKeyword(kw);
                }
                DialogResult result = MessageBox.Show("Record saved successfully.",
"Saved", MessageBoxButtons.OK);
                if (result == DialogResult.OK)
                {
                    this.Hide();
                    ViewReport viewReport = new ViewReport();
                    viewReport.Show();
                }
            }
            else
                label3.Text = "Issue occurred while saving the record. Please try
later.";
        }
```

Reporting Table

```
private void GetListing()
        {
            grdReport.Columns.Clear();
            grdReport.Refresh();

            DateTime? nullDateTime = null;
            int category = cbCategory.SelectedItem != null ? (cbCategory.SelectedItem as
ComboBoxItem).Value : 0;
            DateTime? fromDate = dtpFrom.Checked ? dtpFrom.Value : nullDateTime;
            DateTime? toDate = dtpTo.Checked ? dtpTo.Value : nullDateTime;
            List<ReportListing> listing = new
ReportComponent().GetReportListings(category,
                chkSelfScan.Checked, StaticValues.UserId,
string.IsNullOrEmpty(textBox1.Text) ? null : textBox1.Text,
```

```
                string.IsNullOrEmpty(textBox2.Text) ? null : textBox2.Text, fromDate,
toDate);
            grdReport.DataSource = listing;
            DataGridViewButtonColumn col = new DataGridViewButtonColumn();
            col.UseColumnTextForButtonValue = true;
            col.Text = "View";
            col.Name = "Details";
            grdReport.Columns.Add(col);
        }
```

Start Execution of Software

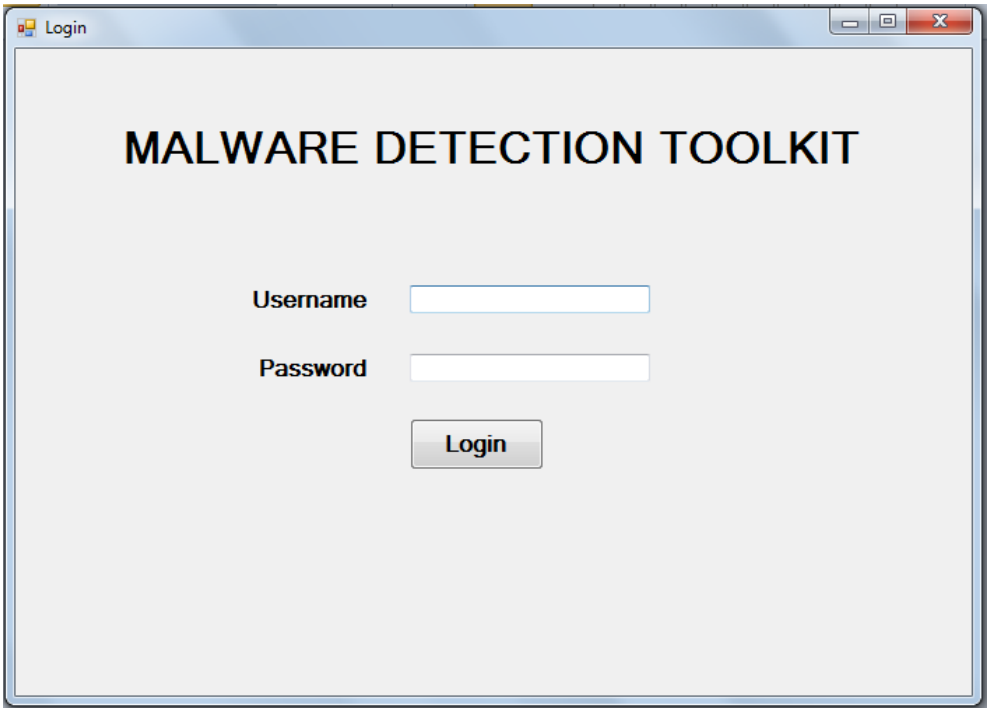
```
    private void button1_Click(object sender, EventArgs e)
        {
            if (string.IsNullOrEmpty(txtTitle.Text))
                return;
            var                    software                    =                    new
SoftwareComponent().GetSoftwarebyId(StaticValues.SelectedSoftwareId);
            ENTITIES.ExecutionLog exe = new ENTITIES.ExecutionLog();
            exe.SoftwareId = StaticValues.SelectedSoftwareId;
            exe.UserId = StaticValues.UserId;
            exe.Startdate = DateTime.Now;
            exe.Title = txtTitle.Text;
            if (new ExecutionComponent().Save(exe))
            {
                button2.Visible = button3.Visible = txtTitle.ReadOnly = true;
                button1.Visible = false;
                StaticValues.ExecutionLogId = exe.ExecutionLogId;
                System.IO.Directory.CreateDirectory(AppDomain.CurrentDomain.BaseDirectory
+ "reports\\" + StaticValues.UserName + "\\" + txtTitle.Text);
                Process.Start(software.Location);
            }
        }
```

Taking Screenshots

```
private void button2_Click(object sender, EventArgs e)
        {
            ScreenCapture sc = new ScreenCapture();
            // capture entire screen, and save it to a file
            Image img = sc.CaptureScreen();
            Bitmap b = new Bitmap(img);
            string        filename        =        DateTime.Now.Year.ToString()        +
DateTime.Now.Month.ToString()          +          DateTime.Now.Day.ToString()          +
DateTime.Now.Hour.ToString() +
                              DateTime.Now.Minute.ToString()                              +
DateTime.Now.Second.ToString() + DateTime.Now.Millisecond.ToString() + ".jpg";
            b.Save(AppDomain.CurrentDomain.BaseDirectory       +       "reports\\"       +
StaticValues.UserName + "\\" + txtTitle.Text + "\\" + filename);
            //       capture       this       window,       and       save       it       -
http://stackoverflow.com/questions/1163761/capture-screenshot-of-active-window
            ENTITIES.ScanScreenShot sss = new ENTITIES.ScanScreenShot();
            sss.ExecutionLogId = StaticValues.ExecutionLogId;
            sss.Image = filename;
            sss.CreatedDate = DateTime.Now;
            new ExecutionComponent().SaveScreenShot(sss);
        }
```

Appendix G- User Manual (Super Admin)

a) **Login Panel:** This includes two user profiles: User Login and Admin Login. Authentication is described based on the rights given to the user type. The Super Admin can login through a default name and password, which will be created during development, and new users (investigators) can be added later by the Super Admin on a rights basis. Functions in report creation, such as read/write, can be used by the Super Admin, while only the read option is given to the user. This logs the users on, and their activity can be viewed by the Super Admin for analysis:



b) **Scan**: This functionality will allow Admin to start any investigation by selecting any particular category and respective tool to execute the system. Once the software is executed and starts running, a new panel for capturing and recording the screen result will pop up. The user/investigator can record the result at any time during the

investigation. Scanning cannot start without adding a title for the investigation. Captured screenshots will be automatically added to the investigation.

c) **Report**: Once the scanning is done, a user can create his or her own standard report. The report section will allow any user to view/edit reports. Edit will allow users to add/delete/modify the description and keywords with the respective investigation (scanning). Super Admin can view/add/delete any user's report while other users can view reports on the basis of access control.

d) **Manage Software:** Super Admin is only allowed this functionality. Through this functionality, users can add/delete/edit any tool category and respective software. Users can also add/delete/modify the path for database storage.

e) **Manage User**: Super Admin is only allowed this functionality. Through this function, users can add any investigator or end user to the list of authentication. They can also activate or deactivate the respective user.